

AD-A022 160

DOD WEAPON SYSTEMS SOFTWARE MANAGEMENT STUDY

A. Kossiakoff, et al

Johns Hopkins University

Prepared for:

Assistant Secretary of Defense  
(Installations and Logistics)

June 1975

DISTRIBUTED BY:

**NTIS**

National Technical Information Service  
U. S. DEPARTMENT OF COMMERCE

OR6140

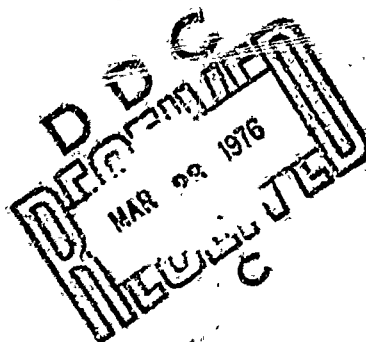
APL/JHU

SR 75-3

JUNE 1975

ADA022160

# DOD WEAPON SYSTEMS SOFTWARE MANAGEMENT STUDY



THE JOHNS HOPKINS UNIVERSITY • APPLIED PHYSICS LABORATORY

Approved for public release; distribution is unlimited.

REPRODUCED BY  
NATIONAL TECHNICAL  
INFORMATION SERVICE

REPORT DOCUMENTATION PAGE		
1. REPORT NUMBER APL/JHU SR 75-3	2. GOVT ACCESSION NO	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DOD WEAPON SYSTEMS SOFTWARE MANAGEMENT STUDY		5. TYPE OF REPORT & PERIOD COVERED Special Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) A. Kossiakoff, T. P. Sleight, E. C. Prettyman, J. M. Park, and P. L. Hazan		8. CONTRACT OR GRANT NUMBER(s) N00017-C-72-4401
9. PERFORMING ORGANIZATION NAME & ADDRESS The Johns Hopkins University Applied Physics Laboratory Johns Hopkins Rd. Laurel, Md. 20810		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Task 2C-6
11. CONTROLLING OFFICE NAME & ADDRESS Office of the Assistant Secretary of Defense (Installations and Logistics), B. C. DeRoze Pentagon, Rm 2A318, Washington, D.C. 20360		12. REPORT DATE June 1975
		13. NUMBER OF PAGES 194
14. MONITORING AGENCY NAME & ADDRESS Naval Plant Representative Office 8621 Georgia Ave. Silver Spring, Md. 20910		15. SECURITY CLASS. (of this report) Unclassified
16. DISTRIBUTION STATEMENT (of this Report)  Approved for public release; distribution unlimited.		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE  na
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) <u>Computer software</u> Development support software and facilities      Modular software architecture Disciplined programming      Operational software Hierarchy of program elements      Software acquisition guides (cont'd)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)  This report to the Office of the Secretary of Defense summarizes the findings of a Weapon System Software Management Study and presents 17 recommendations directed to the alleviation of the more serious problems encountered. The first part of the report contains a discussion of the study objectives and a summary of the recommendations including suggested implementations. Subsequent sections present material from which the recommendations were derived: a review of 10 major previous DoD studies, reviews of 10 Navy and 2 Army Weapon Systems, and summaries of discussions with service organizations and industrial software system contractors. Further discussions of the recommendations and identification of areas requiring further investigation are presented. The report concludes with a guide to five separately published appendices (SR 75-3A, -3B, -3C, -3D, and -3E).		

APL/JHU  
SR 75-3  
JUNE 1975

# **DOD WEAPON SYSTEMS SOFTWARE MANAGEMENT STUDY**

THE JOHNS HOPKINS UNIVERSITY ■ APPLIED PHYSICS LABORATORY  
Johns Hopkins Road • Laurel, Maryland • 20810  
Operating under Contract N00017-72-C-4401 with the Department of the Navy

Approved for public release; distribution is unlimited.



Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

19. KEY WORDS (cont'd)

Computer software (cont'd)

- Software test bed
- Software configuration items
- Software definition
- Software deliverables
- Software vs hardware tradeoffs
- Software visibility
- Structured Programming
- Test and integration software and facilities
- Test and validation tools
- Top-down design

Computer software management

Recommendation areas:

- Management policy
- Acquisition planning
- Systems engineering
- Implementation procedures
- Program management support
- Acquisition management standards
- Development of tools and techniques

Computer systems

- AN/UYK-7
- AN/UYK-20
- AYK-10
- Bendix BDX 820
- Burroughs D84
- CDC 5400B
- CP-642/USQ-20
- CP-642B/USQ-20
- CP-789/UYK
- CP-848/UYK
- CP901/ASQ-114 (V)
- Litton L-304F
- Mk 157
- Raytheon WCC
- Teledyne CP-1050

Conference reports, computer

- Air Force Logistics Command Operation Flight Program Support
- ASAP Ad Hoc Committee for Army Tactical Data System Software Development
- Automatic Data Processing Costs in DoD
- CCIP-85
- Electronics-X Study
- Government/Industry Software Sizing and Costing Workshop
- Monterey Symposium on the High Cost of Software
- Proceedings of the Aeronautical Systems Software Workshop
- Project Pacer Flash
- Tactical Computer Software Acquisition and Maintenance Staff Study

Weapon Systems

AEGIS	DLGN-38	Pershing
CV	E-2C	S-3A
DDG-9	F-14	SAM-D
DLG-28	P-3C	Trident

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

# ABSTRACT

The Applied Physics Laboratory of The Johns Hopkins University (APL/JHU), in conjunction with the MITRE Corporation, carried out a study of Weapon Systems Software Management under the auspices of the Office of the Secretary of Defense. The study was coordinated and supported by a DoD Software Management Steering Committee that included representatives of the Army, Navy, Air Force, and Marine Corps as well as the sponsoring offices of the Department of Defense. Study information was acquired in three parts: (a) review of 10 recent major DoD-sponsored studies that were found to contain a comprehensive body of factual material and expert opinion on virtually all aspects of System Software; (b) review of the software design and management of 10 Navy and 2 Army Weapon Systems; and (c) discussions with service and industry organizations involved in Weapon System software acquisition, development, and maintenance. This information was analyzed and used to generate 17 major recommendations that are presented in this report. The report itself is organized into eight sections as follows: Section 1 summarizes objectives, the study approach, and software problem areas; Section 2 contains summary statements of 17 near-term recommended actions, identifying problems addressed and specific implementation; Section 3 contains a short review of each of the 10 previous DoD studies; Section 4 summarizes the findings of the Weapon Systems reviewed in the APL study, grouped by type of platform; Section 5 summarizes discussions held with service organizations and industrial software system contractors; Section 6 elaborates on the recommendations summarized in Section 2, providing more depth and identifying sources of supporting data from previous studies, Weapon System reviews, or discussions with services or industry; Section 7 identifies a number of areas of potential payoff that require further investigation in order to derive well-founded recommendations for additional specific actions; and Section 8 provides a guide to the separate appendix volumes to this report.

ACCESSION for	
NTIS	White Section <input checked="" type="checkbox"/>
DDC	Blue Section <input type="checkbox"/>
UNCLASSIFIED	<input type="checkbox"/>
CLASSIFICATION	
BY	
DISTRIBUTION AND AVAILABILITY CODES	
CLASSIFICATION OF SPECIAL	
A	

## CONTENTS

List of Illustrations . . . . .	ix
Preface . . . . .	xi
Acknowledgment . . . . .	xiii
1. Introduction . . . . .	1-1
1.1 Objectives . . . . .	1-1
1.2 Approach . . . . .	1-1
1.2.1 Review of Previous DoD Studies . . . . .	1-1
1.2.2 Weapon System Studies . . . . .	1-2
1.2.3 Discussions with Services and Industry . . . . .	1-2
1.3 Problem Areas in Software Life Cycle . . . . .	1-2
1.4 Report Organization . . . . .	1-4
2. Summary of Recommended Actions . . . . .	2-1
2.1 Management Policy . . . . .	2-3
2.2 Acquisition Planning . . . . .	2-6
2.3 Systems Engineering . . . . .	2-8
2.4 Implementation Procedures . . . . .	2-11
2.5 Program Management Support . . . . .	2-14
2.6 Acquisition Management Standards . . . . .	2-17
2.7 Development of Tools and Techniques . . . . .	2-19
3. Review of Previous Studies . . . . .	3-1
3.1 Electronics-X: A Study of Military Electronics with Particular Reference to Cost and Reliability . . . . .	3-3
3.2 Tactical Computer Software Acquisition and Maintenance Staff Study . . . . .	3-4
3.3 Report of the Army Scientific Advisory Panel Ad Hoc Committee for Army Tactical Data System Software Development . . . . .	3-6
3.4 Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980's (CCIP-85) . . . . .	3-7
3.5 Project Pacer Flash . . . . .	3-8
3.6 Automatic Data Processing Costs in the Defense Department . . . . .	3-10
3.7 A Report on Air Force Logistics Command Operation Flight Program Support . . . . .	3-11
3.8 Proceedings of the Aeronautical Systems Software Workshop . . . . .	3-13
3.9 Proceedings of a Symposium on the High Cost of Software Held at the Naval Postgraduate School, Monterey, California, on September 17-19, 1973 . . . . .	3-14
3.10 Government/Industry Software Sizing and Costing Workshop . . . . .	3-15

4.	Highlights of Weapon System Studies	4-1
4.1	Introduction . . . . .	4-1
4.2	Shipborne Systems . . . . .	4-4
4.2.1	DLG-28 Guided Missile Frigate . . . . .	4-8
4.2.2	DDG-9 Guided Missile Destroyer . . . . .	4-12
4.2.3	DLGN-38 Guided Missile Frigate . . . . .	4-16
4.2.4	AEGIS Weapon System . . . . .	4-20
4.2.5	Aircraft Carrier (CV) Tactical Data System . . . . .	4-24
4.3	Airborne Systems . . . . .	4-28
4.3.1	E-2C Tactical Data System . . . . .	4-32
4.3.2	P-3C Airborne Patrol System . . . . .	4-36
4.3.3	S-3A Airborne Weapon System . . . . .	4-40
4.3.4	F-14 Avionics and Weapon Delivery System . . . . .	4-44
4.4	Undersea and Landbased Systems . . . . .	4-48
4.4.1	Trident Command and Control System . . . . .	4-50
4.4.2	Pershing Weapon System . . . . .	4-54
4.4.3	SAM-D Weapon System . . . . .	4-58
5.	Supplementary Discussions with Service and Industrial Organizations . . . . .	5-1
5.1	Service Organizations . . . . .	5-1
5.1.1	Fleet Combat Direction Systems Support Activities . . . . .	5-2
5.1.2	Rome Air Development Center . . . . .	5-5
5.1.3	Center for Tactical Computer Sciences . . . . .	5-6
5.1.4	Joint Logistics Commanders' Software Reliability Work Group . . . . .	5-7
5.2	Industrial Contractors . . . . .	5-11
5.2.1	TRW Systems Group . . . . .	5-13
5.2.2	System Development Corporation . . . . .	5-16
5.2.3	Boeing Aerospace Company . . . . .	5-19
5.2.4	USC Information Sciences Institute . . . . .	5-21
5.2.5	GRD, Inc. . . . .	5-24
5.2.6	General Research Corporation . . . . .	5-25
6.	Discussion of Recommendations . . . . .	6-1
6.1	Management Policy . . . . .	6-4
6.1.1	Analysis and Validation of System Requirements . . . . .	6-4
6.1.2	Software Visibility in Weapon Systems Acquisition . . . . .	6-9
6.1.3	Software as Contract Deliverable . . . . .	6-11
6.2	Acquisition Planning . . . . .	6-13
6.2.1	Milestoned Development Plan . . . . .	6-13
6.2.2	Computer System Resource Development Plan . . . . .	6-17

THE JOHNS HOPKINS UNIVERSITY  
APPLIED PHYSICS LABORATORY  
LAUREL, MARYLAND

6.3	Systems Engineering . . . . .	6-19
6.3.1	Systems Engineering of Computer Systems . . . .	6-19
6.3.2	Provisions for Growth in System Requirements . . . . .	6-22
6.3.3	Systems Engineering of Computer Software . . . .	6-25
6.4	Implementation Procedures . . . . .	6-29
6.4.1	Software Development Support Tools and Facilities . . . . .	6-29
6.4.2	Disciplined Programming . . . . .	6-33
6.4.3	System Integration and Test Capability . . . .	6-36
6.5	Program Management Support . . . . .	6-41
6.5.1	Technical Staffing of Program Manager Organization . . . . .	6-41
6.5.2	System Engineering Agent . . . . .	6-43
6.5.3	Software Operational Support Agent . . . . .	6-45
6.6	Acquisition Management Standards . . . . .	6-47
6.6.1	Standard Criteria for Weapon Systems Computer Resources Acquisition Management . . . .	6-47
6.6.2	Software Acquisition Guides . . . . .	6-49
6.7	Development of Tools and Techniques . . . . .	6-53
6.7.1	Software Test Tools . . . . .	6-53
7.	Subjects for Further Investigation. . . . .	7-1
8.	Guide to Appendices . . . . .	8-1
8.1	Findings and Recommendations of Previous Studies . . . .	8-1
8.2	Weapon System Studies . . . . .	8-2
8.3	Bibliography . . . . .	8-3
	References . . . . .	R-1

Enclosure (1)

Memorandum to Assistant Secretaries for I&L and for R&D of the Army, Navy, and Air Force entitled Management of Weapon Systems Software.

# LIST OF ILLUSTRATIONS

1-1	Interrelation of Software Acquisition Study Findings . . . . .	1-3
4-1	Comparison of Shipborne Systems . . . . .	4-5
4-2	DLG-28 Combat System . . . . .	4-8
4-3	DDG-9 Combat System . . . . .	4-12
4-4	DLGN-38 Combat System . . . . .	4-16
4-5	AEGIS: Projected Ship System . . . . .	4-20
4-6	CV Tactical Data System . . . . .	4-24
4-7	Comparison of Airborne Systems . . . . .	4-30
4-8	E-2C Tactical Data System . . . . .	4-32
4-9	P-3C Airborne Patrol System . . . . .	4-36
4-10	S-3A Airborne Weapon System . . . . .	4-40
4-11	F-14 Avionics and Weapon Delivery System . . . . .	4-44
4-12	Trident Command and Control System . . . . .	4-50
4-13	Pershing Weapon System . . . . .	4-54
4-14	SAM-D Weapon System . . . . .	4-58
5-1	JLC Digital System Definitions . . . . .	5-8
5-2	ARPA Network, Logical Map, January 1975 . . . . .	5-22
6-1	Development of Requirements and Design Criteria . . . . .	6-6
6-2	MIL-STD-483/Milestone Document Correlation Matrix . . . . .	6-15
6-3	Bottom-Up Versus Top-Down Software Development . . . . .	6-27
6-4	Support Software in Software Engineering . . . . .	6-30
6-5	TRW/JSC Program Utilizes Software Development Tools for all Phases of Development . . . . .	6-32
6-6	Operational Computer System Integration and Test Facility . . . . .	6-37
6-7	Partial System Life Cycle Model . . . . .	6-50

## PREFACE

This report contains the substance of the Applied Physics Laboratory's study of DoD Weapon Systems Software Management as submitted to the DoD Software Management Steering Committee. Although limited time was available for the study, every effort was made to define specific and feasible courses of action backed by a coherent rationale and referenced to supporting experience in Weapon System programs and/or expert opinion. It is hoped that these recommendations will serve as a valid basis for the subsequent implementation phase.

The appendices to this report are under separate cover; their scope is outlined in Section 8.

## ACKNOWLEDGMENT

The Applied Physics Laboratory wishes to thank the many individuals and organizations that provided guidance, information, expert advice, and other assistance during the course of this study. Full cooperation was provided by the Weapon System Program Managers of the 12 major programs reviewed, as well as by the contractors and service organizations involved with software development and maintenance. Industrial organizations visited provided detailed information on their software management procedures, advanced developments, and experiences in software acquisition. Valuable discussions were held with staff members responsible for software management policy in the three services. Particular appreciation is expressed to Rear Admiral D. A. Webster, Colonel R. D. Hensley, Jr., and B. DeRoze for their guidance, as well as to all members of the DoD Software Management Steering Committee.

This main report was prepared by A. Kossiakoff, T. P. Sleight, E. C. Prettyman, J. M. Park, and P. L. Hazan with assistance from T. J. Keen and K. R. Wander. The editor for the main report and the associated appendices was H. M. Stainer. The APL staff members contributing to the study of the individual Weapon Systems, the analysis of the previous DoD-sponsored studies, and the establishment of the Software Library are acknowledged in the appropriate appendices. The assistance of the APL Publications Group in preparing these documents in a short time frame is greatly appreciated.



## 1. INTRODUCTION

### 1.1 OBJECTIVES

The Johns Hopkins University Applied Physics Laboratory (APL/JHU), in conjunction with the MITRE Corporation, carried out a study of the management of Weapon System software under the auspices of the Office of the Secretary of Defense. The objectives of this study were stated in a joint letter (Ref. 1, included herein as Enclosure (1)) from the Director of Defense Research and Engineering, the Assistant Secretary of Defense (I&L), and the Assistant Secretary of Defense (Comptroller), dated 3 December 1974, as being "to identify and define (1) the nature of the critical software problems facing the DoD, (2) the principal factors contributing to the problems, (3) the high payoff areas and alternatives available, and (4) the management instruments and policies that are needed to define and bound the functions, responsibilities and mission areas of weapon systems software management." The study was coordinated and supported by a DoD Software Management Steering Committee, chaired by RADM D. A. Webster OASD (I&L). The Steering Committee membership included representatives of the Army, Navy, Air Force, and Marine Corps as well as the sponsoring offices of the Department of Defense.

### 1.2 APPROACH

In accordance with plans developed at the onset of the study, the acquisition of information was conducted in three parts:

1. Review and analysis of ten recent major DoD-sponsored studies that relate directly to objectives of the present DoD Software Study.
2. Review of the software design and management in ten Navy and two Army Weapon Systems.
3. Discussions with service and industry organizations involved in Weapon System software acquisition, development, and maintenance.

#### 1.2.1 Review of Previous DoD Studies

A major objective of this study was to build on the results of the substantial and representative body of informed opinions that resulted from the recent major DoD-sponsored studies of various aspects of software acquisition. These documents were found to contain a comprehensive body of factual material and expert opinion on virtually all aspects of System Software which provided an excellent foundation for the present study. In particular, the studies were used to define the principal problems stated to exist in the acquisition of Weapon System software, and to generate a "road map" of cause/effect relationships between them. In Section 6 of this report the recommended actions resulting from this study are related to the problem area "road-map."

### 1.2.2 Weapon System Studies

The second and most important part of the study concerned specific applications of software design and management to major Weapon Systems. It was agreed that APL would concentrate on Navy systems, MITRE on Air Force systems, and each would study several Army systems. This report describes the highlights of the review of ten Navy and two Army systems assigned to APL.

The objectives of the survey of individual Weapon Systems were first, to serve as a basis for understanding Weapon System software management in the user environment; second, to distinguish among the large range of uses of software in Weapon Systems, and the way these differences affect software problems; third, to provide a working picture of the organizational relationships among Government program managers, system contractors, software contractors, and Government support maintenance and training facilities; and fourth, to identify the design and management techniques that have proven most successful and appear to warrant more general application.

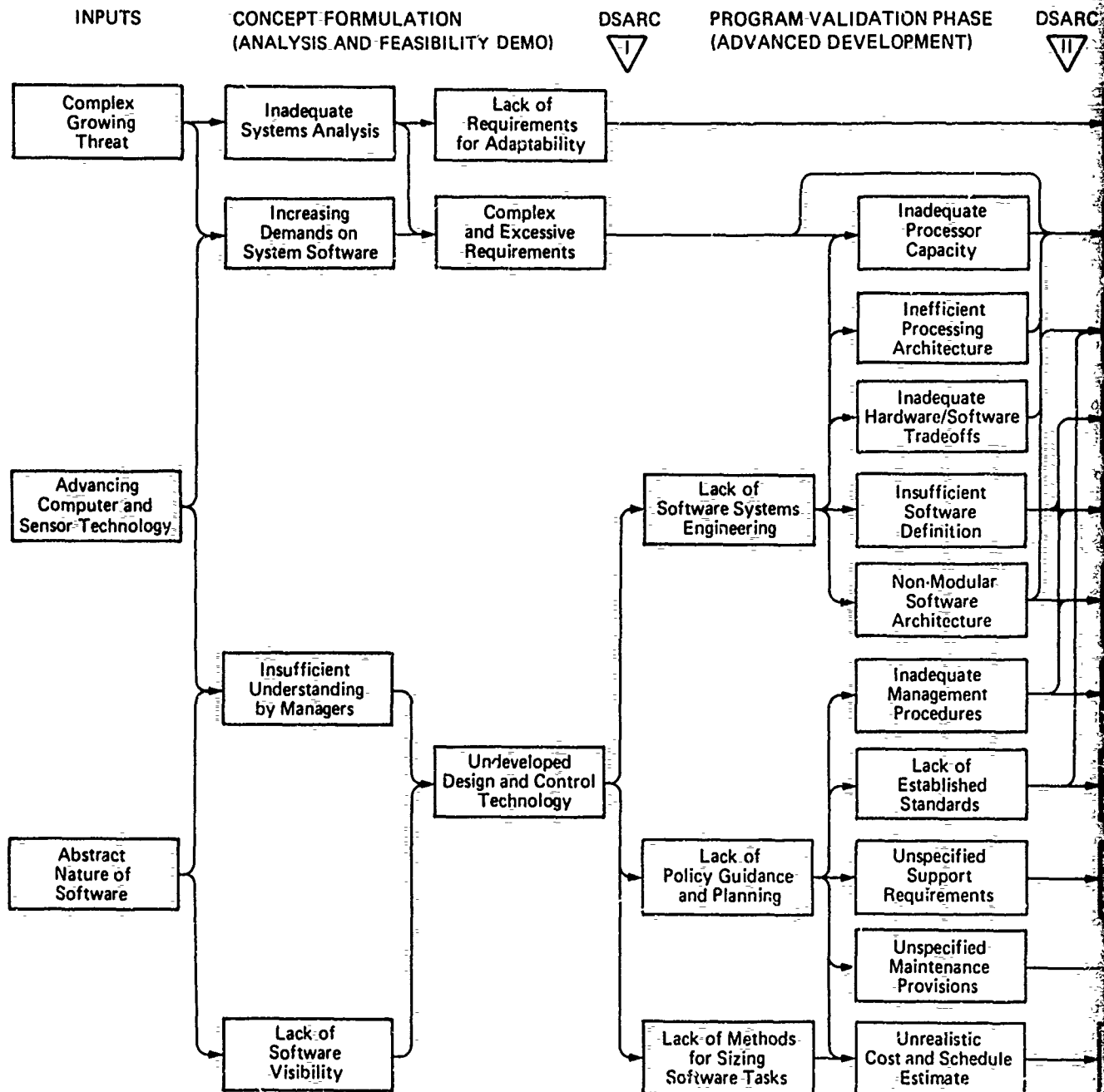
### 1.2.3 Discussions with Services and Industry

In the third part of the study, meetings were held with representatives of DoD software management offices, support facilities, and R&D centers. In addition, visits were made to a number of software system contractors experienced in various aspects of software development and management techniques. These discussions were conducted to establish a data base of existing software standards, practices, and developments, as well as to obtain a wide range of expert opinion concerning current problems in software acquisition management and promising approaches to their correction.

It should be noted that the scope and time of the study did not permit the APL study team to visit a number of major service organizations and leading industrial software contractors. However, it is believed that the results obtained are representative and sufficient to support the recommendations made in this report.

## 1.3 PROBLEM AREAS IN SOFTWARE LIFE CYCLE

As stated earlier, in attempting to understand the nature of the critical software problems facing the DoD and the principal factors contributing to these problems, it was found helpful to develop a "road map" in which problem areas most frequently cited in previous reports on this subject were related to one another and to the principal phases of the software life cycle. The diagram that was developed is shown in Fig. 1-1. Each block in the figure corresponds to a problem area cited in several reports, and its location along the horizontal axis corresponds to the phase in the life cycle, as shown at the top of the figure, in which the problem is first manifest. The occurrence of the principal Defense Systems Acquisition Review Council (DSARC) reviews is also indicated at the top of the figure. The flow lines



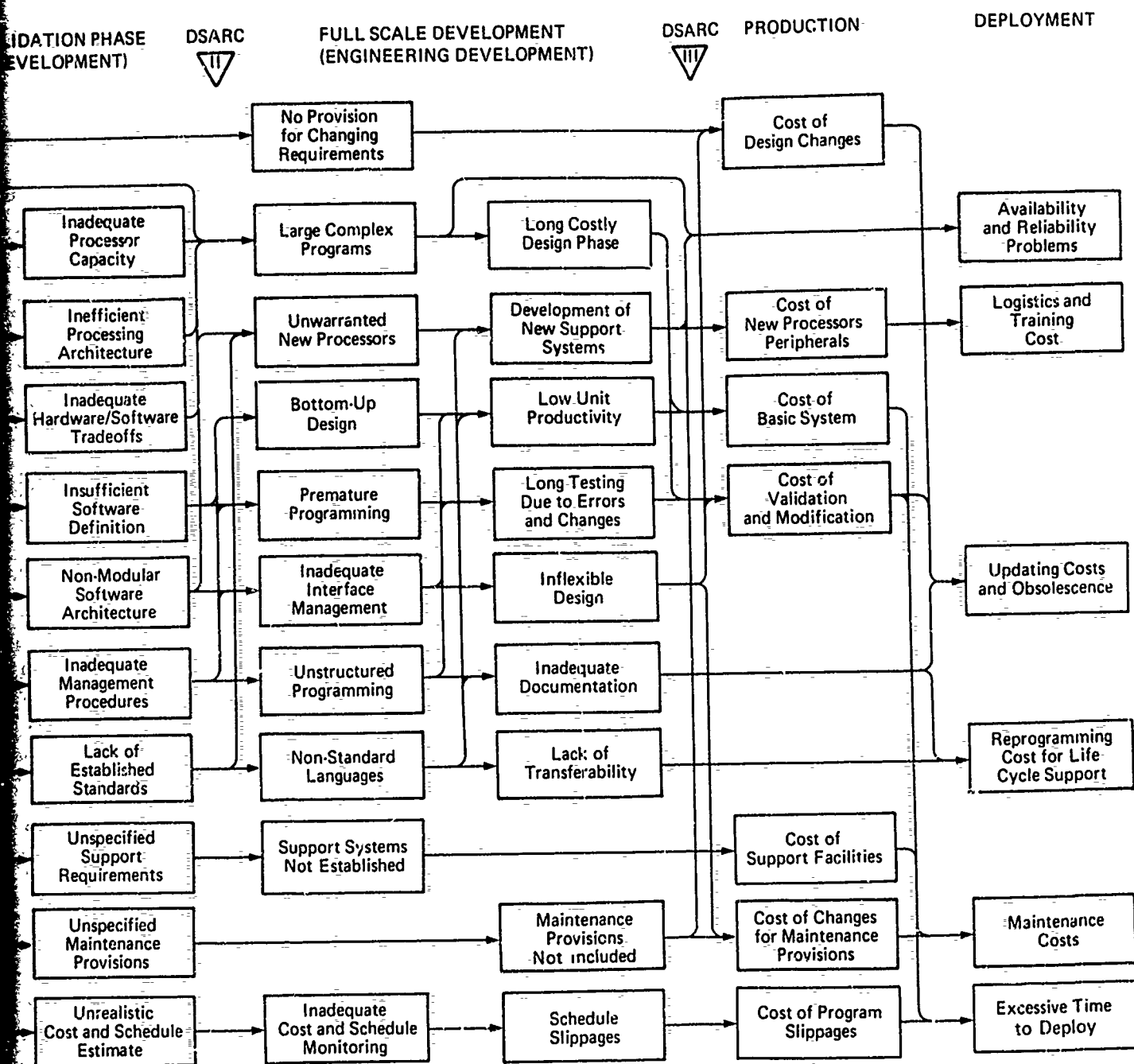


Fig. 1-1 Interrelation of Software Acquisition Study Findings

B

are intended to represent cause/effect relationship among the problems identified.

The column headed "Inputs" represents basic factors contributing to the problems that develop during concept formulation and subsequent phases. The two primary driving factors are: (1) the complex growing threat, which presents an ever-increasing Weapon Systems performance challenge; and (2) advances in computer and sensor technology, which present expanding opportunities to meet these challenges by means of more sophisticated systems. Neither of these factors is subject to restraints. The resulting increase in demands on system software must be recognized as a fact of life that requires major improvements in software design and management technology to keep the situation from becoming even more difficult.

The third basic input is seen to be the abstract nature of software, which in turn contributes to generally underdeveloped design and control technology. It is in this area, together with firm control of changing system requirements, where the greatest improvements in management of Weapon System software acquisition may be achieved.

The large number of problems that arise during the later stages of Program Validation and during Full Scale Development is seen to be the result of the combined impact of heavy and growing requirements on system performance on the one hand and the relatively undeveloped state of software design and control methodology on the other. The manifestations of these problems are numerous. In the recommendations that follow most of the suggested approaches lie either in the area of software systems engineering or policy guidance and planning, which are different but related approaches to the development of improved software design and control methodology.

An important factor that must be considered is identified in the flow line at the top of the figure, which represents aspects of the inevitable growth and change of requirements throughout a system's lifetime. These are inherent in the changing nature of the threats against which most systems must operate, as well as in the fact that software can be modified without physical changes to the system. In practice, unless provisions for adaptation to change are designed into a system, the consequences are often serious.

#### 1.4 REPORT ORGANIZATION

This report summarizes the findings made in the three main parts of the APL study, and describes a number of recommended actions directed to the alleviation of the more serious problems encountered in the management of Weapon System software, as well as to the exploitation of areas deemed to have highest payoff for management action.

The main report is organized in eight sections, as follows:

Section 1 (above) summarizes objectives, the study approach, and software problem areas.

Section 2 contains summary statements of 17 near-term recommended actions, identifying problems addressed and specific implementation.

Section 3 contains a short review of each of the 10 previous DoD studies.

Section 4 summarizes the findings of the Weapon Systems reviewed in the APL study, grouped by type of platform.

Section 5 summarizes discussions held with service organizations and industrial software system contractors.

Section 6 elaborates on the recommendations summarized in Section 2, providing more depth and identifying sources of supporting data from previous studies, Weapon System reviews, or discussions with services or industry.

Section 7 identifies a number of areas of potential payoff that require further investigation in order to derive well-founded recommendations for specific action.

Section 8 provides a guide to the Appendices to this report.

## 2. SUMMARY OF RECOMMENDED ACTIONS

This section contains concise statements of specific actions designed to attack many of the problems encountered in recent years in the development and support of software for major Weapon Systems. These are grouped under seven categories, as follows:

### Management Policy

- MP1 Analysis and Validation of Systems Requirements
- MP2 Software Visibility in Weapon System Acquisition
- MP3 Software as Contract Deliverable

### Acquisition Planning

- AP1 Milestoned Development Plan
- AP2 Computer System Resource Development Plan

### Systems Engineering

- SE1 Systems Engineering of Computer Systems
- SE2 Provisions for Growth in System Requirements
- SE3 Systems Engineering of Computer Software

### Implementation Procedures

- IP1 Software Development Support Tools and Facilities
- IP2 Disciplined Programming
- IP3 System Integration and Test Capability

### Program Management Support

- MS1 Technical Staffing of Program Manager Organization
- MS2 Systems Engineering Agent
- MS3 Software Operational Support Agent

### Acquisition Management Standards

- AM1 Standard Criteria for Weapon System Computer Resources  
Acquisition Management
- AM2 Software Acquisition Guides

### Development of Tools and Techniques

- TT1 Software Test Tools

A fuller discussion of the problems and recommended actions in each category is given in Section 6, Discussion of Recommendations.

The recommended actions were developed as a consensus of ideas drawn from previous studies, from field investigations of ten Navy and two Army Weapon Systems, and from interviews with a number of major industrial software contractors and service organizations. Most recommendations are supported by several independent sources.

Every effort has been made to propose specific remedial actions, by relating each action either to a particular type of document that could be amended or amplified, or to an existing review process. Because of the limited scope and time available to conduct this study it has been impossible to verify that each proposed action is the best and most practical means of effecting the desired objective. For this reason, it is hoped that a follow-up effort will be made to establish the most effective measures to achieve near-term solutions to the problems addressed. If this is done, the proposed actions should constitute a useful point of departure.

It may be noted that while the implementing actions proposed are definite, the agency to carry out the actions is often not specified. This has been done purposely in the belief that the specific organization of official action on these recommendations should be determined by the DoD Software Management Steering Committee and not by the study teams.

In phrasing the recommended implementations, most of the terms referring to management procedures are the same as those initially defined by the Joint Logistics Commanders' Software Reliability Work Group (SRWG) and developed further by the Technology Coordinating Panel of the DoD Software Management Steering Committee, or employed in the recently issued Air Force Regulation 800-14 (Ref. 2); they are presented in Subsection 5.1.4 of this report.

It will be noted that there is only one recommendation listed under "Development of Tools and Techniques." This is because the recommendations included in this section are only those that are appropriate for attention at the DoD level and for which definite near-term implementing action was devised. A number of other significant R&D areas were found to fail at least one of the above criteria and are discussed elsewhere in this report.

Each recommended action will be discussed under four headings. The "Recommendation" section gives a brief general statement of the proposed action. The particular difficulties that the recommended action is designed to alleviate in present software acquisition are listed under "Problems Addressed." A statement of the specific action recommended, with reference to particular management documents or reviews, is given under "Implementation." The "Remarks" section provides additional discussion of the action recommended and, occasionally, supporting references.



## 2.1 MANAGEMENT POLICY

### ANALYSIS AND VALIDATION\* OF SYSTEM REQUIREMENTS

MP-1

#### Recommendation

Direct that a comprehensive analysis and definition program be carried out on software (as well as hardware) elements of each new major Weapon System during the Program Validation Phase, prior to approval of Full Scale Development. The software definition should be carried down to the level of subprograms performing major functions.

Cost estimates for the development and integration of each subprogram should be based on analysis, simulation, modeling, or construction of its principal parts, as called for by their respective newness or criticality.

#### Problems Addressed

The fact that requirements often change during system development is generally acknowledged to be one of the greatest sources of cost escalation and schedule slippage in the acquisition of major software systems. Moreover, initial requirements are often unduly ambitious, imposing very difficult or impossible design conditions that often lead to changes in the design approach. These changes arise in large measure because the initial requirements were not critically analyzed and validated through a formal program of advanced development or system definition.

Despite the implications in the DSARC II review that an adequate design and costing basis must exist, current directives are vague on the formal requirements for the validation phase of the acquisition process. Many software cost overruns that stem from vague and inconsistent requirements could be eliminated by more thorough analyses and reviews of requirements specifications.

#### Implementation

Amplify and extend the definition of requirements for the System Design Review (SDR) to include review of the functional definition of each software program, its modular configuration, and its evaluation in terms of core and timing allocations in order to show that it will meet performance requirements within specified constraints (similar to Milestone 1 in SSD 61-47B, Ref. 3).

Amplify the requirements of the DSARC II review in DoD-5000 (Ref. 4) with respect to what is expected to be accomplished in the Program Validation Phase.

#### Remarks

In order to properly validate requirements of new (as opposed to evolutionary) systems and to establish a realistic basis for estimating costs, it has been necessary to perform a limited amount of preliminary design, as well as paper analysis. In particular, critical design features that cannot be adequately modeled must be built and tested to ensure that a feasible design approach exists. Thus, the implementation of this recommendation requires the accomplishment of limited preliminary design during the Program Validation Phase, rather than as the first step of Full Scale Development.

In addition, in order to eliminate vagueness, each requirement should be checked for testability.

\*"Validation" is used in the context of Program Validation Phase, as opposed to validation/verification of coded computer programs.

SOFTWARE VISIBILITY IN WEAPON SYSTEM ACQUISITION

MP2

Recommendation

Increase the visibility and understanding of major software components of Weapon Systems by putting them on a par with hardware components. This is to be done in terms of configuration control items, LSARC reviews, design reviews, and other aspects of acquisition management.

Problems Addressed

The lack of software visibility, as compared to hardware, in the acquisition of major subsystems is generally agreed to contribute to the fact that it is not as well managed. This acquisition management problem, in turn, results in numerous sins of omission throughout the development process that result in unrealistic cost and schedule estimates, inadequate configuration management, and related problems.

Implementation

One specific action that should be taken is to generalize and revise Appendix B (Electronics System) of Military Standard-Work Breakdown Structures for Defense Material items (MIL-STD-881, Ref. 5) to define a software subsystem category on a par with a hardware subsystem along the lines being advocated by the Joint Logistics Commanders' Software Reliability Work Group (SRWG). Construct a suitable list of computer resources that should be covered during review processes. Other directives and standards for hardware should be studied by the DoD to find similar transferability of their principles to software.

Remarks

The Joint Logistics Commanders' SRWG is proposing a revision to MIL-STD-881 to call out software subsystems at the proper level in the work breakdown structure, as well as other actions to give software more visibility. It is recommended that their efforts be reviewed, supported, and applied to all services.

The Tactical Computer Software Acquisition and Maintenance Staff Study (Ref. 6) and the Pacer Flash study (Ref. 7) had similar recommendations.

## SOFTWARE AS CONTRACT DELIVERABLE

MP3

### Recommendation

Specify that major computer software involved in Weapon Systems development be designated Configuration Items (CI's) and deliverables during Full Scale Development. This would generally include computer programs and computer data for

1. Operational Software,
2. Development Support Software, and
3. Test and Integration Software.

### Problems Addressed

The absence of clear definitions and guidelines applicable to the components of software (computer programs and computer data) as distinguished from software documentation has caused numerous instances where Weapon System contracts have not called for appropriate items as deliverables. The same lack of guidance has limited software visibility and hence the effectiveness of configuration management. In many systems where operational software is defined as a contract deliverable, support software needed for system maintenance has not, and had to be redeveloped. In other instances new support software has been developed when tools available from other programs could have been used. Practices also vary concerning the procurement of integration and test software.

### Implementation

Provide clear guidelines for designating appropriate computer system resources (computer programs and computer data) as CI's in the Program Management Directives and manuals. Call for scheduled delivery, like hardware items. Specify support and test and integration software as separate deliverables. Present directives are vague in this regard.

### Remarks

It is generally agreed that the formalization of software as contract deliverable is a desirable and cost-effective practice for insuring effective life cycle support of major systems. The F-14 Program did not specify the operational software as a contract deliverable and has had difficulty in obtaining a suitable product after the fact. The Trident Program treats the operational software and integration and test software as deliverables. Discussions at Boeing indicated that this was also the case for the B-1 avionics development. The P-3C and S-3A Programs have followed this practice.

In many of the systems surveyed, the support and test software (including assemblers, compilers, libraries, simulators, test tools, and data reduction programs) exceed the size of the operational programs. In addition, software for on-line equipment operability testing and fault isolation (Automatic Test Equipment, ATE) has become important in modern systems. In the AEGIS system, where this is a deliverable, the on-line test program for an operational ship is projected to be several times the size of the tactical programs.

As with other deliverables, provisions must be made for appropriate acceptance tests for all classes of software. Software documentation should be examined to ensure that it adequately explains the software logic and how to maintain the software CI's.

## 2.2 ACQUISITION PLANNING

### MILESTONED DEVELOPMENT PLAN

AP1

#### Recommendation

Define the requirements for milestones in the Full Scale Development phase to ensure the proper sequence of analysis, design, implementation, integration, test, and review processes. Also, define criteria that will be used to demonstrate that each milestone has been achieved.

#### Problems Addressed

The abstract nature of software makes it difficult to measure progress and, hence, makes it even more necessary to formalize the steps in design, implementation, and test. The lack of such definition leads to difficulties in interface management and to the late discovery of inadequate requirements or design errors, with resulting slippages in schedules and increases in cost.

#### Implementation

Amplify the definition of requirements for Preliminary Design Review (PDR) and Critical Design Review (CDR) to specify the items of analysis, design, implementation, integration, and testing to be completed. Develop an updated version of the milestone definition of SSD 61-47B (Ref. 3) or its equivalent. (Note that Milestone 1 of SSD 61-47B should precede Full Scale Development.) Incorporate these in the Program Management Plan and specify that the milestone provisions be written into development contracts.

#### Remarks

Many modern programs have successfully established their own system of milestones, as was found in discussions with the Program Managers for the SAM-D, AEGIS, Trident, S-3A, CV, Pershing, and B-1 systems. In the B-1 program, the milestones contained in SSD 61-47B were written into the contract. Currently applicable Acquisition Management regulations, such as MIL-STD-490 (Ref. 8), MIL-STD-483 (Ref. 9), and AFR-800-14, Vol. II (Ref. 10), call out milestones but do not define the work to be accomplished and the products to be delivered. It is important that each milestone be associated with a specifically defined deliverable.

It is believed that a standard set of milestones, as in SSD 61-47B, is desirable as a common basis for system acquisition, in order to ensure a properly structured and sequenced development approach. The establishment of appropriate milestones in the Request for Proposal (RFP) would also provide a common basis for program planning and hence for proposal evaluation. SSD 61-47B provides an acceptable basis for current procurements but should be updated to include test events and to reconcile event and document-oriented milestones. In so doing, reference should be made to FCDSSA Handbook for Program Development and Production Procedures of Digital Processor Program, H(A)-4010 (Ref. 11).

## COMPUTER SYSTEM RESOURCE DEVELOPMENT PLAN

AP2

### Recommendation

Ensure provision of a detailed Computer System Resource Development Plan as part of the bid package on Full Scale Development contracts. The plan should cover all aspects of the contractor's approach to organization, design, test, management, documentation, and other aspects of the program.

### Problems Addressed

In order to ensure that the development of a major software subsystem is well organized and managed, and all requirements are properly understood and defined, it is essential to have a detailed development plan prepared and evaluated prior to starting Full Scale Development.

The development plan should include a detailed statement of the contractor's engineering and management approaches, and hence can serve as a basis for selecting a contractor with the requisite understanding, experience, and facilities.

### Implementation

Require that the Program Manager prepare a set of development requirements to be included in the Request for Proposal (RFP). Specify those aspects that are directed by the Government. Specify the nature and scope of description required in the contractor's Computer System Resource Development Plan.

### Remarks

AFR-800-14, Vol. II (Ref. 10), calls out a set of detailed requirements for a Computer Program Development Plan. The plan identifies the actions needed to develop and deliver computer program Configuration Items and necessary support resources. It is to be prepared by the implementing command or, if the development effort is contracted, it may be prepared by the contractor and approved by the implementing command.

The plan addresses such items as organization, management controls, design, test and Quality Assurance methodology, program milestones, status monitoring, supporting resources, documentation, and engineering practices. For a full listing of the items covered, see Section 6, Discussion of Recommendations.

SDC and several other contractors stated that a formal development plan is the most important single management document and recommended that it be specifically required for major software systems. Since practice varies widely in Weapon System acquisition, the Computer System Resources Development Plan needs to be recognized as a standard requirement.

## 2.3 SYSTEMS ENGINEERING

### SYSTEMS ENGINEERING OF COMPUTER SYSTEMS

SE1

#### Recommendation

For systems involving several distinct functions, require that the system be divided into functional segments in accordance with the operational requirements. Require during the Program Validation Phase that tradeoff analyses be performed for hardware versus software (i.e., hardwired versus programmable functions) and for different computer system architectures.

#### Problems Addressed

The lack of application of systems engineering methodology to computer system design is at the root of a number of critical problems in the development of major Weapon Systems. It results in inefficient processing architecture, lack of hardware/software tradeoffs, and overcentralization, leading to overly complex requirements and hence large, cumbersome, and costly software programs.

#### Implementation

Call for these tradeoff analyses to be performed during the Program Validation Phase and presented at System Design Review (SDR) and DSARC II.

#### Remarks

Recent advances in low cost digital logic and microprocessors provide the means for unburdening the general purpose computers from high data rate processing functions (e.g., radar data processing) with resulting simplification of processing and executive functions. By the same token, development of minicomputers has made it much easier to subdivide the total computing task into relatively independent major functions performed by dedicated processors, thereby simplifying interface management and providing greater flexibility to accommodate changing requirements. Discussion with the Fleet Combat Direction System Support Activities indicates that such decentralization of noncombat direction functions would substantially simplify the overall data system.

In order to ensure that tradeoff studies are properly conducted, it is necessary to include consideration of system growth potential and management flexibility.

## PROVISIONS FOR GROWTH IN SYSTEM REQUIREMENTS

SE2

### Recommendation

Provide for growth and change in requirements on Weapon System computer software by identifying parameters that are uncertain or are likely to change in the future and, where possible, specify the probable limits on such changes. Also identify novel environments and use of new techniques. Require that computer systems be sized to provide for uncertainties and requirement growth.

### Problems Addressed

The nature of Weapon Systems is such that the inevitable growth and change in the enemy threat, as well as advances in sensor and weapon technology, result in corresponding growth and change in system requirements throughout the life of the system. While, in principle, changes in software should be less expensive than those in hardware, such changes can actually be extremely costly unless provision for growth and change have been made in the initial design. Also, opportunities for designing-to-cost are frozen out unless provision is made for growth.

### Implementation

Require (in appropriate regulations) that the design requirements or specifications that are given to the software developer contain parameter uncertainty limits. Specify provisions to accommodate parameter changes in the Preliminary Design Review (PDR) process. Provide for software breadboarding (including cost scheduling) in novel environments or when using new techniques. Provide in the requirements that substantial reserves of time and memory be delivered; where possible they should be tentatively allocated to specific potential growth requirements. Check all these items at the PDR and at the DSARC II and III reviews.

### Remarks

The history of Weapon System development is replete with examples of costly changes incurred because of overburdened processors. They have been cited in numerous papers and noted by studies such as Electronics-X (Ref. 12) and the Army Scientific Advisory Panel report (Ref. 13), both of which recommend special attention to adequate sizing of computer hardware. Of the Weapon Systems studied, the E-2C and P-3C growth margins were more than absorbed by increased data processing requirements. In the Model 4 update, most NTDS ships are installing a large shared memory module to relieve the overburdened central processors. The Navy has established a policy (TADSTAND-5, Ref. 14) of providing a 20% spare memory, time, and input/output capacity at system delivery. Spare time capacity is even more critical than spare core because it is harder to expand.

The designation of parameter ranges in the system requirements is a procedure found useful in the Site Defense Program. The practice is recommended in order to permit a more flexible organization of the software, with parameters incorporated in modules that can be readily changed or expanded without reflecting into the entire system.

SYSTEMS ENGINEERING OF COMPUTER SOFTWARE

SE3

Recommendation

Specify the use of modular software architecture and an orderly, phased design approach for developing major computer programs that defines the higher levels of the program and then progresses to design and test successively lower levels. The latter approach is often referred to as "top-down" design. It involves the formal definition of a hierarchy of program elements and restrictions concerning lateral communications.

Problems Addressed

The lack of application of systems engineering methods to the design of software has led to systems that are nonmodular, lacking well-established interfaces, and difficult to test. The design approach has often been undisciplined, with implementation started before the overall structure has been defined. This results in incompatibilities and errors that are discovered late in the test process, with serious impact on schedules and costs. Lack of modularity results in complex interfaces and difficulties in accommodating to changes in requirements.

Implementation

Specify the use of modular top-down design in the Request for Proposal (RFP) and use adherence to this approach as a criterion in examining the contractor's definition of his organization of design, production and test, design methodology, and engineering practices.

Remarks

The importance of organizing computer programs into a set of top-level functional modules with defined interfaces is generally accepted as an important principle of software systems engineering. It is also widely accepted that a disciplined, phased approach to software design (usually called top-down design) should be a mandatory requirement. While top-down design is frequently associated with "structured programming," it is quite different; one relates to design and the other to implementation. Top-down design requires that any given function be first designed and then implemented. Because of its hierarchical organization it permits a lower level element in one sequence to be implemented while a higher level element in another sequence is being designed. This approach permits the complete design, implementation, and testing of all the critical paths early in the development, using dummy elements to simulate the noncritical portions.

Top-down design is not incompatible with other development techniques, such as "Threads" used by CSC. Threads was initially developed primarily for function tracing, test, and evaluation and can be used to effectively augment top-down design.

Important advantages of the top-down approach to design and test are that it simplifies the test software, uncovers errors early in the design cycle, and permits one to "build a little, test a little." It does not necessarily apply to the design of software breadboards or utility programs.



## 2.4 IMPLEMENTATION PROCEDURES

### SOFTWARE DEVELOPMENT SUPPORT TOOLS AND FACILITIES

IP1

#### Recommendation

Ensure that the Full Scale Development program includes provision of adequate modern support tools and facilities, including such items as assemblers, compilers, editors, debug aids, data base and library management systems, and associated operating systems. Require maximum use of existing proven tools and facilities. Provide that any of these tools and facilities that will be required by the Operational Support (Maintenance) Agent for system maintenance be delivered in transferable form and also be capable of application to future Weapon System programs.

#### Problems Addressed

The development of software requires a major investment in support tools and facilities. If they are not available from previous programs and are not provided for in the development plan, a major schedule slippage and cost overrun can result. If they are not designed to be transferable to the Operational Support Agent, as required for system maintenance support, additional costs will be incurred during the maintenance phase. Inadequate support tools lead to excessive testing times and late detection of errors.

#### Implementation

Specify support tool and facility requirements in the Request for Proposal (RFP) and evaluate proposals on the adequacy of existing and planned support facilities and tools, and on the contractor's experience in their use. Have the Operational Support Agent participate in defining the requirements. Define support software as a Configuration Item. Provide that the portion of support software needed for operational maintenance be a contract deliverable with formal documentation. Make provision for support planning in Acquisition Management regulations, and subject it to design review procedures. Provide O&M funds for measures to ensure transferability. Provide for development of new tools in a manner directly transferable to other programs. Continue support and provide funding for the work of the Technology Coordinating Panel of the DoD Software Management Steering Committee on a software catalogue.

#### Remarks

A number of basic development support tools (such as assemblers, compilers, and test drivers) are essential to any program. Provisions should be made for them in the Computer System Resource Development Plan. In recent years a number of computer aids have been developed to assist in various phases of program design, implementation, and testing. These include data base management systems, automated logic testers, and other development and test aids. Depending on the tools available at Operational Support facilities from previous programs, a portion of the support tools should be provided as deliverables under the development contract. This has not been consistently done in major system acquisitions.

In contracting for a major software development it is most important to ensure that the selected contractor has the requisite experience and capability with modern software development tools. This factor should be a major consideration in contractor selection.

## DISCIPLINED PROGRAMMING

IP2

### Recommendation

Require that the computer program development contractor apply a highly disciplined set of engineering practices to the detailed design and programming phases of development. This must involve a clear and disciplined set of standards covering program structure, size, control, interface, formal conventions on data base management, and the demonstration that the standards are enforced in practice.

### Problems Addressed

The design of software traditionally has been a craft rather than an engineering discipline. Consequently it has tended to be unstructured, with few rules and constraints. In large scale real-time systems involving many designers and programmers, an undisciplined approach has frequently resulted in software that is difficult to integrate, debug, and update. This, in turn, leads to excessive costs in testing and life-cycle maintenance.

### Implementation

The Request for Proposal (RFP) should call for a description of the contractor's design and coding manuals and his approach to programming discipline in the Computer System Resource Development Plan. Formal and well-established procedures that have been demonstrated on prior programs should be an important element in the contractor selection process. The contract should specify that the proposed procedures be used.

### Remarks

Most experienced software development contractors have a well-established and documented set of programming standards. These define a discipline for logical design, data conventions, calling conventions, interface control, and other important aspects of program structure. It is important to obtain and evaluate these standards in the process of contractor selection.

A particular technique for disciplining logical design, called "Structured Programming," has been promulgated recently. The technique has gained considerable support and is now required in several programs. However, the extent of its appropriate application has not yet been fully established, especially in the case of real-time systems involving input/output and other time-sensitive functions. What is important is that logical design be rigidly controlled by one means or another, and that control flow be a visible and traceable characteristic of the program documentation. Such discipline is currently practiced by experienced software contractors.

## SYSTEM INTEGRATION AND TEST CAPABILITY

IP3

### Recommendation

Require that an integration and test capability be provided as part of Full Scale Development of major Weapon System software, tailored to the specific needs of the program. This should be a software test-bed combining simulated elements and hardware (including operator consoles) to be used in progressive integration and test of system elements. It should provide real-time dynamic stimuli and responses under repeatable and off-nominal test conditions. The portion of this capability that is required for Operational Support and Maintenance should be specified to be transferable or capable of duplication.

### Problems Addressed

In many cases, the first confrontation of a Weapon System with a realistic environment occurs when it is installed aboard the operational platform and subjected to operational testing. Under these circumstances, major performance deficiencies are often discovered too late to remedy without basic changes to the system software. In addition, interface incompatibilities are encountered that require time-consuming and costly changes. Further, even operational testing does not readily subject the system to the extreme conditions that constitute its design limits.

### Implementation

Define the provision of an integration and test capability as a requirement in the Request for Proposal (RFP) and in the Computer System Resource Development Plan. Specify that the portion of the simulation software required for system operational support and maintenance be made a contract deliverable with formal documentation. Provide O&M funds to the contractor for support of maintenance features. Constrain sophistication to avoid overcomplication, especially at the contractor facility. Make provisions for Integration and Test Facility planning in Acquisition Management regulations, and subject such planning to design review procedure. Consider training requirements for test facilities.

### Remarks

It is generally agreed that simulation, test, and integration tools are required throughout the life of major combat systems. This is particularly well recognized in avionic systems, where the flight environment must be simulated during the basic design of the operational software and where the real-time interaction of operators and displays constitutes an essential design factor. All avionics systems reviewed had a "hot-bench" or equivalent facilities for progressively tying in and dynamically testing software/hardware modules as an integral part of the development and integration efforts.

It is important that the specific scope and nature of simulation/integration/test tools be tailored to the functional nature of the system. It is easy to overdo the realism and hence add excessive complexity and cost to this type of system.

The integration and test capability should be carefully planned and strictly controlled to provide the appropriate facility during each phase of the program. For operational support, the system integration facility may typically contain much of the actual system hardware. However, it is still necessary to use simulated inputs to provide controlled and reproducible test conditions, especially inputs that stress the design limits of the system.

## 2.5 PROGRAM MANAGEMENT SUPPORT

### TECHNICAL STAFFING OF PROGRAM MANAGER ORGANIZATION

MS1

#### Recommendation

Establish and implement a policy that Program Managers for major Weapon Systems be staffed with personnel experienced in systems engineering and software development and of sufficient stature and number to carry out essential management functions that cannot be delegated.

#### Problems Addressed

A wide variation exists in the degree to which Program Managers are staffed with personnel competent in systems engineering and software application. Adequate staffing is essential, particularly in the early stages of a program (concept formulation, validation, and contract definition) during which many basic technical decisions must be made. Such decisions as the selection of applicable standards should be made with an understanding of design and cost impact.

After initiation of Full Scale Development, sufficient support is needed to assure competent technical design review and system test monitoring.

#### Implementation

Provide for high level review (e.g., DSARC I and II) of Program Manager staffing at the start of Program Validation and the Full Scale Development Phases of major Weapon System development programs. Provide means for temporary assignment of engineers from service laboratories and support activities to fill key staff positions. Provide career incentives to attract competent engineers from within and from outside the Government into both military and civilian positions. Establish policies that assure adequate grade levels for Civil Service jobs in this area.

#### Remarks

The proper technical staffing of program management organizations must take into account the nature and scope of the program, the resources available within the headquarters organization, the support organizations available to the Program Manager, the type of contract, and other management considerations. However, in most instances it will be necessary to take special steps to augment the Program Manager's staff from within the Government. For this reason it is important to develop ways in which qualified persons throughout the service organization could be assigned to key positions in the program management organization. Special management attention is needed to create sufficient priority to effect such assignments. The problem of career incentives for military personnel may be more severe in some services than in others, so that special attention is required to ensure that appropriate action will be taken.

SYSTEMS ENGINEERING AGENT

MS2

Recommendation

Establish a policy that, for major new Weapon System programs, the Program Manager engage a Systems Engineering Agent to assist in problems arising in the translation of system requirements into detailed hardware and computer system design requirements. The agent, whether Government or contractor, should be highly experienced in system operational requirements, special purpose system hardware, and computer system software and hardware.

Problems Addressed

Although the Program Manager should have on his immediate staff systems engineers who are knowledgeable about software, manpower limitations often restrict the staff to a skeleton organization. Without other direct support, the Program Manager cannot adequately fulfill his responsibilities for carrying out the extensive planning and monitoring associated with a major new Weapon System. This can result in insufficient definition of requirements, limited requirements analysis, unrealistic schedule and cost estimates, and inadequate configuration management.

Implementation

Include identification of a Systems Engineering Agent in the Program Management Plan, the Program Management Directive, and the Computer System Resource Development Plan. Provide for the agent's participation and, in appropriate cases, leadership in Concept Formulation and Program Validation, as well as in preparation for DSARC reviews, in the Request for Proposal process, and in Preliminary and Critical Design Reviews (PDR and CDR).

Remarks

Many large-scale Weapon System programs use technical organizations in the role of the Systems Engineering Agent. The E-2C Program Manager (as well as several other Program Managers) uses the NAVAIR Computer and Software Systems Group (Code 533) for technical support and review. In the area of Ship Combat Direction Systems, FCDSSA itself functions in this capacity. For the AEGIS program, APL serves as a technical advisor to the Program Manager. TRW performs such a function for the Minuteman System. Technical support should be tailored to the specific needs of a particular program.

SOFTWARE OPERATIONAL SUPPORT AGENT

MS3

Recommendation

Require that the Software Operational Support (Maintenance) Agent be identified and consulted during the Program Validation Phase to support the Program Manager in providing for maintenance support requirements. Require that the agent be included at the beginning of and throughout Full Scale Development to plan for system integration, testing, and transfer from development to operational status.

Problems Addressed

The integration of operational support requirements and the transition from production into operational use are high on the list of major problems in Weapon Systems acquisition. The lack of transferability of software, the lack of provisions for maintenance, and the cost of changes resulting from these inadequacies have been cited in many previous software studies as important problems needing solution.

Implementation

Amplify those parts of the Program Management Plan and the Program Management Directive dealing with the early participation of the Using and Supporting Commands to include the identification of an Operational Support Agent. Provide means for applying O&M funds to support contractor activity directed toward providing maintenance capabilities and documentation.

Remarks

This recommendation is widely supported by Weapon System managers. The main problem is one of implementation. It requires an early decision as to how and by whom the operational maintenance task is to be performed on a new system. Many systems are currently being developed without a designated Operational Support Agent. Top management action is required to ensure that a decision is made at an early date.

A study was recently made by SDC for the Air Force Logistics Command (Refs. 15 and 16) on the assignment of Operational Support responsibility for a number of avionic systems. The recommendations were made on a case-by-case basis, but nearly all called for an activity managed by the Government and supported by a contractor, usually the one responsible for system development. This suggests that policy attention should also be directed toward providing contractor support after system deployment. This would preserve the knowledge of detailed system design for use during system maintenance and updating.

## 2.6 ACQUISITION MANAGEMENT STANDARDS

### STANDARD CRITERIA FOR WEAPON SYSTEM COMPUTER RESOURCES ACQUISITION MANAGEMENT AM1

#### Recommendation

Establish a common set of requirements and criteria to be applied in the acquisition and support of Weapon System computer resources by all services.

#### Problems Addressed

Many of the preceding recommendations have proposed policies with regard to various aspects of system acquisition management. Their implementation requires the establishment of one or more top-level documents that would constitute official guidance to Program Managers and contracting officers. Current MIL-STDs on this subject are not adequate and are primarily hardware oriented. Variation in terminology is another problem that must be addressed to reduce confusion and the misinterpretation of existing guidelines.

#### Implementation

Derive a tri-service document covering the procedures to be used in the acquisition and support of Weapon System computer resources, using current service regulations and manuals as a basis. Such a document would most appropriately become a Military Standard. It is suggested that Air Force Regulation (AFR) 800-14, Vol. II (Acquisition and Support of Computer Resources in Systems, Ref. 10) be used as a point of departure, amended as recommended herein and to be consistent with MIL-S-52779(AD) (Software Quality Assurance Program Requirements, Ref. 17) and NAVMATINST 4130.1A (Configuration Management, Ref. 18). Additional material may be drawn from the Army's AMC Pamphlet AMCP-70-4 (Software Acquisition - A Guide for the Material Developer, Ref. 19) and the Navy's Computer Software Management Task Force Document Outlines (Ref. 20). Use a common terminology along the lines recommended by the Joint Logistics Commanders' Software Reliability Work Group.

#### Remarks

Current Military Standards dealing with System Acquisition Management are MIL-STD-499A (USAF) Engineering Management (Ref. 21) and MIL-STD-483 (USAF) Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs (Ref. 9). Neither addresses the special problems of software acquisition.

The Air Force has recently issued AFR-800-14 (Refs. 2 and 10), a relatively comprehensive treatment of the management planning, engineer management, and configuration management aspects of software acquisition. While quite detailed, it leaves reasonable flexibility to accommodate conditions of individual programs. This document appears to be the best available starting point for a common set of criteria on software acquisition management.

MIL-S-52779(AD), prepared by the Army, provides very recent guidance on Quality Assurance and is generally not in conflict with AFR-800-14. It is being used in several Navy programs.

These documents, and recent Army and Navy efforts referred to above, would appear to constitute a good starting point for a common document. The participation of industry in the preparation of such a document is recommended.

## SOFTWARE ACQUISITION GUIDES

AM2

### Recommendation

Prepare a series of handbooks or guides covering important aspects of software acquisition, to help Program Managers and their staffs to define, review, and evaluate requirements, procedures, proposals, and designs during pre-contract and contract management. These would include such items as

- Life Cycle Plan
- System Requirements Review
- RFP Preparation and Review
- Computer Resource Development Plan Review
- Preliminary and Critical Design Reviews
- Documentation Standard Selection
- Support Facility Plan Evaluation
- QA Plan Evaluation

### Problems Addressed

The great variation in the requirements and structure of Weapon Systems, differences between new and evolutionary systems, different methods of contracting, and organization of the sponsoring agency all require a large degree of flexibility in the application of management standards and procedures. However, the abstract nature of software and the relatively underdeveloped systems engineering methodology make it very difficult for Program Managers and their limited staffs to apply the necessary judgment in the absence of an organized body of knowledge to guide them.

### Implementation

Coordinate current service efforts or assemble a tri-service committee with government and industry representation, under the sponsorship of OSD, to prepare suitable handbooks. Issue drafts for interim guidance and to obtain feedback from experience. Allocate special funds to participating service agencies.

### Remarks

There are efforts in progress by all the services to prepare manuals and handbooks on various aspects of software acquisition management and documentation. While each service has somewhat different problems, the similarities are greater than the differences, and considerable advantage would accrue from coordinating or unifying these efforts. Since the Joint Logistics Commanders' Software Reliability Work Group is also proposing the preparation of a series of handbooks, their recommendations should be considered in any action on this matter.



## 2.7 DEVELOPMENT OF TOOLS AND TECHNIQUES

### SOFTWARE TEST TOOLS

TT1

#### Recommendation

Support development of improved software test and validation tools to reduce the cost and time involved in software verification. These should include automated tools to identify and exercise all branches, to detect and isolate design faults, and to categorize error sources.

#### Problems Addressed

Test and validation has been the most time-consuming phase of software development. This has been true not only because of the numerous errors introduced by poor design methodology but also because of the effort required to design test drivers for individual portions of the program. In addition, manual testing of the full range of possible input conditions (in order to exercise all portions of the program) is extremely time consuming. Finally, the generation and running of test programs are subject to human error, which further adds to the validation time.

#### Implementation

Support ongoing service programs in development of automated test and validation tools. Fund the conversion of selected tools to the high level languages used in Weapon Systems (e.g., CMS-2, JOVIAL) and provide them to system contractors and Operational Support Activities as soon as economically practicable. Invite innovative proposals for new work. Support R&D efforts in software portability to aid in the application of tools to different systems.

#### Remarks

All of the industrial software contractors who were interviewed considered software test and validation one of the significant payoff areas for decreasing software costs. Most of these contractors had made substantial in-house efforts in the development of automated test tools and a number were using such tools in software development programs. Automated test case generation is an area that particularly deserves support.

Little is currently known about the root causes of faults in design implementation, primarily because emphasis has been on debug and recovery techniques rather than on analysis. The exploratory work being done in this area by RADC should be supported.

Special attention is required to determine how proprietary test tools can be made more generally available. Consideration should be given to arrangements such as licensing or purchase by the Government.

### 3. REVIEW OF PREVIOUS STUDIES

In the first phase of the DoD Software Management Study four major objectives were specified. Of these, the first two, namely, to identify and define "(1) the nature of the critical software problems facing the DoD," and "(2) the principal factors contributing to the problems," have been the subject of a number of major DoD-sponsored studies and workshops during the past three years. Accordingly, one of the first steps was for the DoD Software Management Steering Committee (established by Office of the Secretary of Defense memorandum dated 3 December 1974, Ref. 1) to identify those studies that are relevant to the subject of Weapon System Software and designate them Baseline Documents. Table 3-1 lists the selected study references. The study contractors were directed to analyze the various study findings in order to expand the base of informed judgment currently available. Figure 1-1, derived from the study review, provides a graphic representation of the phases of a Weapon System life cycle and the chronological relationship of software life-cycle management problems. While the assignments are somewhat arbitrary, the figure permits tracing the consequences of problems arising at early stages in the process to related difficulties in subsequent stages.

This chapter contains a short review of each of the ten Baseline Documents. The results from each of the study or workshop efforts are discussed in terms of the scope and nature of the study, the types of conclusions and recommendations, and the relevance to the conclusions of this report. Because of the diversity in the stated purposes and objectives of each study or workshop, a wide range of conclusions and recommendations was generated. The major conclusions and recommendations are included in Appendix A, Findings and Recommendations of Previous Studies.

In general, there exists a common theme to all the study and workshop documents which points to the need to manage Weapon Systems software in such a manner as to reduce costs and provide greater visibility throughout the acquisition/life-cycle process.

While it would be interesting to know which recommendations from these studies have had an effect on the software management practices of DoD, such correlation has not been attempted. It is, of course, likely that much of this wisdom has been recognized and that many actions have been taken to implement helpful ideas.

For convenience in distinguishing between this report and the previous studies reviewed, the DoD Software Management Study is referred to as "DSS" throughout this section.

TABLE 3-1

BASELINE DOCUMENTS - STUDIES AND WORKSHOPS

Title	Sponsor	Date
Electronics-X: A Study of Military Electronics with Particular Reference to Cost and Reliability	ARPA, DDR&E	JAN 74
Tactical Computer Software Acquisition and Maintenance Staff Study	OSD(I&L)	OCT 73
Army Scientific Advisory Panel Ad Hoc Committee for Army Tactical Data System Software Development	Army	OCT 74
Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980s (CCIP-85)	AFSC	APR 72
Project Pacer Flash	AFLC	SEP 7
Automatic Data Processing Costs in the Defense Department	DDR&E	OCT 74
Air Force Logistics Command Operation Flight Program Support	AFLC	DEC 74
Proceedings of the Aeronautical Systems Software Workshop	AFSC	APR 74
Proceedings of a Symposium on the High Cost of Software Held at the Naval Postgraduate School	AFSC, ARO, ONR	SEP 73
Government/Industry Software Sizing and Costing Workshop	ESD	OCT 74

### 3.1 ELECTRONICS-X: A STUDY OF MILITARY ELECTRONICS WITH PARTICULAR REFERENCE TO COST AND RELIABILITY

#### 3.1.1 Scope

The Electronics-X (Ref. 12) study program, which produced a report of 429 pages, was conducted by the Institute for Defense Analyses (IDA) with the assistance of representatives of industry, private research organizations, and Government. The study was initiated because of such symptoms as "rising acquisition costs, poor field reliability of military systems, and shrinking quantities of weapons." Its specific purpose was to review the "process of acquisition and maintenance of military electronics and recommend specific policies and procedures to remedy the situation. The magnitude of military electronics, its pervasive nature, and its rapid growth led to its being singled out as an area in which massive savings might be achieved."

The study took into account broad principles recommended by earlier investigations and sought specific data that would indicate approaches to a reduction in the costs of electronics acquisition and support consistent with the role of military electronics.

"This report is concerned with three kinds of costs: development, production, and support. Empirical evidence suggests that, statistically, production and support costs are positively correlated; but that development effort can be applied to reduce either one or the other or the sum of the two." Because support costs occur in future years and are neither accounted for by the project manager nor paid for out of current funds, management emphasis is on development and production costs, even though lifetime support costs may dominate.

The Electronics-X report concentrates on five major, high-impact areas of the military electronics acquisition process: (a) data collection and feedback, (b) requirements, (c) competition and management options, (d) reliability enhancement, and (e) maintenance training. Numerous other areas are discussed in the report, and detailed recommendations are made for each.

#### 3.1.2 Conclusion

This report is a valuable reference because the study considers a wide range of factors in the total system acquisition process.

The basic software recommendations are: (a) thorough system design before starting implementation, (b) careful consideration of system architecture, (c) adequately sized processors, (d) disciplined programming, (e) attention to program structure and efficiency, (f) use of standard higher order languages (HOL's) whenever possible, and (g) deferment of coding until the computer design is complete.

The DSS study concurs with most of these recommendations, but differs from Electronics-X with regard to the level of detail specified in item (f) above.

### 3.2 TACTICAL COMPUTER SOFTWARE ACQUISITION AND MAINTENANCE STAFF STUDY

#### 3.2.1 Scope

The objective of the Tactical Computer Software Acquisition and Maintenance Staff Study (Ref. 6), which produced a report of 56 pages plus appendices, was to identify significant problems related to software acquisition and maintenance. The need for this effort became apparent during Defense Systems Acquisition Review Council (DSARC) meetings in which there existed a general lack of information and understanding concerning software.

In efforts to make weapons more effective in the modern combatant environment, U.S. military forces are increasingly using and becoming committed to digital computers and their associated software. A selected set of Weapon Systems acquisition programs/projects was surveyed to identify significant problems related to software acquisition and maintenance. The Weapon Systems included the DD-963, LHA, PF, F-14, SAM-D, TACFIRE, AEGIS, E-2B, E-2C, S-3A, F-15, and VAST. The study team reviewed policies and held discussions with many individuals who were associated with the programs or projects. Collected material was organized into the study report.

#### 3.2.2 Conclusion

The following findings regarding tactical software were based on the specified Weapon Systems program surveys.

1. "There is a marked absence of DoD management policy guidance regarding the use of digital computers and software in vital automated tactical systems" even though tactical software embodies military doctrinal procedures for accomplishing combat functions.
2. During early Southeast Asia operations, certain tactical systems performing the same functions for the military services (air control and defense) were unable to exchange information automatically.
3. "In general, system program/project offices became oriented to acquiring electronics hardware and gave little attention to the process of developing the software."
4. "Military services procured a wide variety of tactical computer hardware and languages, which caused the acquisition of a variety of support software."
5. "The vital function of tactical software management (maintenance) during the life of a tactical system - incorporating new and revised tactical doctrine procedures into operating form - was not adequately recognized."

The recommendations, although few in number, were quite broad in scope. They included such items as: (a) educating DoD top level management to become more knowledgeable of the impact of computers and software, (b) reviewing DoD organizational responsibilities regarding policies, etc. of software acquisition,

THE JOHNS HOPKINS UNIVERSITY  
APPLIED PHYSICS LABORATORY  
LAUREL, MARYLAND

(c) issuing policies that will promote more effective tactical software acquisition, use, and maintenance, and (d) establishment by the DoD Materiel Specifications and Standards Board of a software panel to provide more guidance for military standards, etc. These recommendations are generally supported in this DSS report.

### 3.3 REPORT OF THE ARMY SCIENTIFIC ADVISORY PANEL AD HOC COMMITTEE FOR ARMY TACTICAL DATA SYSTEM SOFTWARE DEVELOPMENT

#### 3.3.1 Scope

"An ad hoc group of the Army Scientific Advisory Panel (A.S.A.P.) was organized to study Army Tactical Data Systems Software developments and to recommend actions to provide improved software at lower cost and in shorter time in future tactical systems." A 39-page report was produced (Ref. 13).

"The group was first convened on 13 June 1973 and in that and subsequent meetings received information from Army agencies, from other government activities, and from commercial organizations on the history or prior developments, on requirements for the future, and on new hardware and software technologies."

The study group defined the scope of their study as: "Determine the factors that lead to extensive and complex software and to problems in developing software for tactical data systems, and to recommend practices and useful exploratory efforts to mitigate those difficulties."

#### 3.3.2 Conclusions

As the study discussions began, it became clear "that the solution of tactical system software problems should not be approached solely through software or programming research and development, although certain efforts in that area will prove fruitful." Software problems often originate from a lack of initial system engineering in the overall tactical data system.

"The ad hoc study group's investigations led to findings and recommendations in four major areas:

1. System Design and System Hardware
2. Software Design and Development
3. R&D Related to Software
4. Army Management of Software Development."

In item 1, recommendations were made concerning system architecture, definition of the processes to be performed, evaluation of current systems, interactive development of hardware and software, and adequate hardware capacity. Item 2 included recommendations for early design of software, standardization and use of higher order languages (HOL's), selection of program libraries, application of disciplined programming, and use of a third party advisor. In item 3, R&D recommendations dealt with standard programming languages, standard operating systems, optimized computer architectures, and improved methods of acquiring hardware and software. In item 4, recommendations covered such subjects as developing meaningful and realistic tools for management and documentation, early agreement on language and operating system requirements, agreement by all parties on tasks, evaluation of development progress, specifications and tools for software testing, and the requirement for evolutionary development of software.

The recommendations of the ASAP are generally in accord with those in this DSS report.

### 3.4 INFORMATION PROCESSING/DATA AUTOMATION IMPLICATIONS OF AIR FORCE COMMAND AND CONTROL REQUIREMENTS IN THE 1980s (CCIP-85)

#### 3.4.1 Scope

Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980s (CCIP-85) (Ref. 22), an AFSC Development Planning Study, was completed early in 1972. "The study purpose was to construct an integrated Air Force R&D program for the 1970's which will develop the information processing technology needed to meet the likely Air Force command and control (C&C) information processing requirements of the 1980's." The study was primarily concerned with C&C for Air Force combatant units.

It was concluded in the report that information processing is barely adequate to support Air Force C&C functions and that the major problems are in the software area rather than in computer hardware technology.

#### 3.4.2 Conclusions

The study revealed a number of trends that by the 1980's will

- (a) "Make C&C considerably more important to Air Force roles and operations,"
- (b) "Make C&C much more dependent on information processing technology," and
- (c) "Sharply increase the strain on software technology imposed by C&C requirements." There exists a high degree of visibility for some of the above trends while others are obscure. However, they are all gathering momentum.

The study group produced a list of problems that were arranged generally in sequence as being critical, significant, or appreciable. These problems are stated in more detail in Appendix A, Findings and Recommendations of Previous Studies.

CCIP-85 provided a list of recommendations that are broad in scope. It was recommended that future improvements should be directed to the use of new and improved programming techniques, better management of simulation, certification, and production facilities, and to the setting of goals to improve the various factors that affect software and its use. The CCIP-85 recommendations, as categorized above, generally agree with those presented in this DSS report.



### 3.5 PROJECT PACER FLASH

#### 3.5.1 Scope

Air Force Regulation 20-1 (Ref. 23), dated 4 December 1972, established Project Pacer Flash "for the purpose of conducting an in-depth study of long-range computer software support required for weapon system computers" and required the Air Force Logistics Command (AFLC) to chair the study group. The regulation also required the participation of the Strategic Air Command, the Tactical Air Command, the Materiel Air Command, the Air Defense Command, the Air Training Command, and the Air Force Systems Command. The final report (Ref. 7), consisting of 43 pages plus attachments, was published in September 1973.

"The task group was charged to:

1. Determine the present and projected inventory of programmable computers installed in weapon systems;
2. Review existing policy and procedures pertinent to support requirements;
3. Develop changes to existing policy and procedures or develop new policy and procedures for weapon system computer support;
4. Recommend an Air Force position on management of software support, and;
5. Publish and update new or changed policy and procedures in appropriate directives."

#### 3.5.2 Conclusions

Although the study was aimed at all Air Force Weapon Systems, the investigation, analysis, and conclusions only addressed aeronautical systems. Several facts in this area were identified by the Air Staff advisors as being important and having a bearing on the problem. Software development standards, contractor support costs, software transferability, lack of adequate (in-house) testing and validation, and the need for greater in-house capabilities for software support were listed as important matters to be considered in the study.

The following statements comprise a partial list of relevant conclusions presented in the final report:

1. It is expected that the complexity, scope, and cost of software support will necessarily increase because of the increase in avionic system complexity.
2. "Headquarters USAF has stated the need for expanding the long-range organic capability" for avionic software support.
3. It is essential for hardware and software support to be considered as an integral problem.
4. Computer "software requires configuration management."
5. In order to assure proper development and later support of software, expansion of Air Force policy and procedures must be accomplished to cover the entire life cycle of a Weapon System.
6. "Planning for software support (supportability) must begin during the conceptual phase of system design."

The recommendations from the study are oriented toward a specific subset of Air Force needs and requirements. They concern (a) Air Force (in-house) capability, (b) responsibility for software with Aeronautical System managers, (c) requirements for software support in the Automatic Test Equipment (ATE), Operational Flight Programs (OFP), and Crew Trainer/Simulator areas, (d) continuous inventory of computer hardware, (e) revision of Air Force Specialty Codes, (f) standardization of hardware/software as an integral problem, (g) consideration of hardware/software as an integral problem, (h) designation of ATE to an avionic system, and (i) software as deliverables. With the exception of those areas that deal with specific Air Force requirements and organizations, the Pacer Flash recommendations are in general agreement with those of this DSS report.

### 3.6 AUTOMATIC DATA PROCESSING COSTS IN THE DEFENSE DEPARTMENT

#### 3.6.1 Scope

Paper IDA-P-1046, Automatic Data Processing Costs in the Defense Department (Ref. 24), containing 68 pages, was prepared by David A. Fisher, Institute for Defense Analyses (IDA), in October 1974. As stated, the "paper attempts to provide substantiated estimates of the costs and cost trends of DoD computer software and other ADP activities, and the major components of those costs, on the thesis that a determination of present costs is the necessary first step in deciding on future DoD investments in ADP research and development." It was pointed out that any specific cost items that were disproportionately high should lead to review, and action would be taken in those areas where urgent attention was required. It was further stated that "the estimates derived in this paper provide insufficient basis in themselves for making recommendations on the future course of DoD R&D on ADP, and no such recommendations are made here. The findings, however, provide an estimate of computer software costs to DoD and show the structure of those costs. This information is necessary for developing and evaluating DoD software research programs."

Dr. Fisher's paper acknowledged the CCIP-85 Air Force study (Ref. 22) as "the most extensive analysis available on computer software and ADP in DoD." Dr. Fisher based much of his work on CCIP-85. It should be noted that he was primarily concerned with ADP activities rather than tactical software since CCIP-85 uses are limited to an investigation of large scale automated command and control systems.

#### 3.6.2 Conclusions

Although, as stated above, there are no formal recommendations presented in the ADP cost study, the conclusions and findings point out some important facts that deserve further consideration and will, no doubt, influence decisions in many different phases of the software acquisition process. It was noted that "reliable information on most software and ADP costs in DoD is unavailable in a clearly identifiable form." Cost figures were presented which indicate that software is a major expenditure and a significant part of all electronics costs in DoD. Other findings show the upward trends in the cost of software, the shift from rental to purchased equipment, the upward trend in the number of computing systems, and a shift from using in-house personnel to contract services for systems analysis/design.

3.7 A REPORT ON AIR FORCE LOGISTICS COMMAND OPERATION FLIGHT PROGRAM  
SUPPORT

3.7.1 Scope

In July 1974, System Development Corporation (SDC), under contract to Warner Robbins Air Logistics Center (ALC), initiated a study of the support methodology used by various other ALC's within the Air Force Logistics Command (AFLC). The study considered selected "transitioned" aircraft systems and their respective Operational Flight Programs (OFP's). The study output report included 450 pages plus appendices (Refs. 15 and 16).

During the past several years there has been a major trend toward the use of more digital elements in aeronautical systems. "The software associated with programmable devices, typical to integrated avionic systems, has emerged as one of the key elements affecting avionics and weapon system performance."

The specific objective of the study was to produce a final report that reflected the findings of data collected, an analysis of the data, and recommendations and conclusions formulated from the analysis. The data collected by SDC from the various ALC's were divided into six major sections:

1. Characteristics of the OFP's and their associated aeronautical systems,
2. Current support posture of the OFP's,
3. Personnel required in the support of the OFP's,
4. Documentation required in the support of the OFP's,
5. Configuration management required in support of the OFP's, and
6. Testing in support of the OFP's.

3.7.2 Conclusions

"It has become evident that the Air Force requires a capability to effectively manage contractor developments in the software area which is as effective as the Air Force's capability to manage hardware procurements." Because the Air Force in-house capability to support in-depth software changes is generally limited, it has been necessary to retain the development contractor to provide software support even after transition from the Air Force Systems Command (AFSC) to AFLC.

Some of the basic conclusions of the study effort were (a) "Configuration management practices and procedures were not performed in a standard manner," (b) the Technical Order System does not address the subject of program documentation, (c) in most cases software support was not provided to AFLC for OFP's at transition, (d) "In no case was a condition found that standardization of programming languages, or the rewrite of a program from one system to another, was warranted," and (e) "The possibility of rewriting a computer program from one computer to another is not feasible . . . because of the extreme differences between machines."

The recommendations cover such areas as (a) the establishment of baseline software configurations, (b) support software, (c) introduction of a new specification system, and (d) definition of organizational structure concerning Weapon System managers and functional areas. Of these, the first three directly support the recommendations of this DSS study; the fourth deals generally with Air Force special requirements in the aeronautical systems area.

### 3.8 PROCEEDINGS OF THE AERONAUTICAL SYSTEMS SOFTWARE WORKSHOP

#### 3.8.1 Scope

Because recent studies, such as Project Pacer Flash (Ref. 7), had emphasized and recommended the need to exchange communications at all levels, AFSC Development Plans sponsored an Aeronautical Systems Software Workshop in April 1974 to provide an opportunity to exchange information on avionics systems software. Software had received a great deal of management attention and concern because of costs and lack of visibility as to progress etc. The attendees at the workshop included personnel from the Air Staff, Air Force Systems Command, Air Force Logistics Command, the User Commands, and other Government agencies. Members of industry and DoD presented papers on how they had successfully coped with many problems involved in the avionics systems software acquisition process. The final report of the workshop consists of 576 pages (Ref. 25).

In his opening remarks to the workshop, Major General Douglas T. Nelson set the stage for the presentations and discussions by emphasizing the major importance of software to achieve flexibility for modern airborne Weapon Systems. He pointed out that "accompanying our digital trend with its flexibility has been the increased involvement with the attendant software: all the way from design, development, through test and transition to the Air Force Logistics Command and operation by the using commands."

Another statement by General Nelson: "If any part of our development job ever called for skillful systems management, this task of bringing the whole weapon system together with software certainly does!!!"

#### 3.8.2 Conclusions

No formal set of recommendations resulted from the workshop. The individual papers addressed a wide range of software problem areas, and many solutions were proposed. Certain papers did list specific recommendations, however, and these lend support to those presented in this DSS report.

In general, the following subjects were covered in the discussions or in formal recommendations: (a) disciplined programming, (b) software development standards, (c) total system integration and testing capability, (d) properly trained members of program offices, (e) third party design engineering agent, (f) third party support in the operational area, (g) computer resources acquisition management, (h) software acquisition handbooks or guides, (i) tools for testing software, (j) systems engineering of computer software, (k) total systems engineering, (l) provision for future growth and expansion, (m) development plans, (n) milestone events in the development process, (o) software as a specified deliverable, and (p) provision of software visibility.

Without exception, the recommendations discussed in the individual papers lend support to the recommendations of this DSS study report.

3.9 PROCEEDINGS OF A SYMPOSIUM ON THE HIGH COST OF SOFTWARE HELD AT  
THE NAVAL POSTGRADUATE SCHOOL, MONTEREY, CALIFORNIA, ON  
SEPTEMBER 17-19, 1973

3.9.1 Scope

"The Monterey Symposium on the High Cost of Software was held in September 1973, under the joint sponsorship of the Air Force Office of Scientific Research, the Army Research Office, and the Office of Naval Research." The symposium was called primarily to define the research required to achieve a major reduction in software costs because the software art was progressing so slowly. High software costs and poor quality made a serious impact on the DoD budget and operations. A 138-page report was produced (Ref. 26).

The symposium was divided into five workshops, each of which was concerned with a specific aspect of software. The five workshops were assigned the following themes:

- "Workshop 1 - Understanding the Software Problem
- Workshop 2 - Semantics of Languages and Systems
- Workshop 3 - Programming Methodology
- Workshop 4 - Software-Related Advances in Computer Hardware
- Workshop 5 - Problems of Large Systems."

Three meetings of the symposium as a group were held during which subjects of interest to the entire body were presented. A keynote speech on software costs and statements of objectives by the workshop chairmen were the subjects of the first meeting. The second meeting was devoted to software technology transfer, and the third meeting included interim progress reports by workshop chairmen.

3.9.2 Conclusions

"Over the last ten years there has been a radical shift in the balance of hardware and software costs. Because of technological advances, hardware costs have been reduced to the point where hardware designers are now seeking ways to help reduce software costs. The cost of computing is now clearly dominated by the cost of software." Other conclusions and findings are discussed in detail in the Findings and Recommendations of Previous Studies Appendix to this Department of Defense Software Study (DSS) report.

The recommendations produced by the symposium were quite general in nature and covered a broad spectrum of software acquisition problems. Recommendations dealing with the following subjects were specified: (a) strengthen research in computer systems, (b) enhance the programming aids for a domain with knowledge derived from the related applications domain, (c) employ new techniques from the laboratories, (d) create a new technology base, (e) coordinate service, DoD, and civilian R&D efforts, (f) increase research to meet future demands, (g) develop understanding of software costs, (h) employ the best available programming aids, (i) discipline programming, (j) formulate new concepts for testing and analysis, (k) investigate the human factors in programming, (l) develop new programming methods and system architectures for new applications, and (m) construct better programming languages and computer systems.

Most subjects in the above areas are covered in the recommendations in this DSS study.

### 3.10 GOVERNMENT/INDUSTRY SOFTWARE SIZING AND COSTING WORKSHOP

#### 3.10.1 Scope

The Government (Electronics Systems Division (ESD))/Industry Software Sizing and Costing Workshop was held on 1-2 October 1974 at the ESD of the Air Force Systems Command (AFSC), Hanscom Air Force Base, Massachusetts. There were 76 attendees of whom half were from Government and half from industry. Twenty-one companies, one university, and nine USAF and other governmental units were represented.

The general purpose of the workshop was to seek a means of enhancing the communications between the Government and industry on the problems of predicting software development costs. "More specifically, the workshop focussed attention on two key questions.

"What are the attributes of a good software requirements specification?

"What are the prime factors affecting/driving software costs?"

The ultimate objective was to significantly enhance the realism/credibility of future software costing and sizing estimates for electronics defense systems.

In order to have discussion groups of workable size, the workshop was divided into four groups. These groups addressed the two questions stated above and developed answers that are summarized in the draft report. The draft report was dated 11 February 1975 and consisted of 38 pages plus attachments (Ref. 27).

#### 3.10.2 Conclusions

The conclusions dealt with (a) definition and purpose of specifications, (b) lack of thorough analysis and validations of requirements, (c) lack of goals or tradeoffs in Requests for Proposals (RFP's), (d) ranges of operating conditions in RFP's, (e) tendency to specify design rather than stating performance requirements, (f) separation of design ideas from performance requirements, (g) cost estimating, (h) phased contract approach to software acquisition, and (i) cost-calculated-on-instruction count.

The formal recommendations from the study specified actions in the following areas: (a) a multiphased and/or separate contract approach to software acquisition, (b) improvement of specification standards and practices, (c) standardization of terminology, and (d) improvement of cost estimating for software. All of these recommendations lend support to those in this DSS study report, although estimating software costs is not explicitly recommended.



#### 4. HIGHLIGHTS OF WEAPON SYSTEM STUDIES

##### 4.1 INTRODUCTION

This phase of the study was concerned with specific applications of software design and management to major Weapon Systems. APL concentrated on Navy Systems, MITRE on Air Force Systems, and each contractor studied several Army Systems. The systems were selected to represent a variety of platforms and major missions and to illustrate all phases of the Weapon System life cycle. Of necessity, emphasis was placed on breadth at the expense of depth. It is hoped, however, that the significant highlights in each program, those which can be of value to others, have been uncovered and understood in their proper context.

The survey of individual Weapon Systems, as a major input to this study (DSS), had the following objectives:

1. To serve as a basis for understanding how and what Weapon Systems software is being or has been developed, produced, deployed, and maintained in the user environment;
2. To serve as a basis for distinguishing among the large range of uses of software in Weapon Systems; differences in function, size, and complexity; and the way these differences affect software problems and potential solutions;
3. To provide insight into the organizational relationships between the Government Program Managers, system contractors, software contractors, and Government test, maintenance, and training facilities;
4. To identify design and management techniques that have proved successful and that warrant more general application, and
5. To obtain opinions from key personnel concerning ways in which the OSD or the Services can contribute to the improvement of software cost and performance.

The survey of Weapon Systems software was carried out through the auspices of the respective Program Managers. System and software contractors were visited, where possible, to obtain first-hand information on system characteristics and development methods.

The selected Weapon Systems programs are listed in Table 4-1. The systems are grouped into Shipborne Systems, Airborne Systems, and Undersea and Landbased Systems. Subsequent sections of this report highlight program characteristics pertinent to this study.

More detailed information gathered on each Weapon System is presented in Appendices B, C, and D to this report. The individual discussions vary in detail because of the differing stages of development of the different systems. The following kinds of information were sought:

1. General System Description: A sufficient description to provide understanding of the overall system mission and requirements and the operating environment of the embedded computer system;

TABLE 4-1  
WEAPON SYSTEMS INVESTIGATED

	Weapon System Programs	Systems	Status
Shipborne Systems	DLG-28	Tactical Data System Terrier Weapon System	Deployed
	DDG-9	Tactical Data System Tartar Weapon System	Deployed
	DLGN-38	Command and Control System Sensor Interface Data Sys- tem Tartar Weapon System Gun Fire Control System Underwater Fire Control System	Production
	AEGIS	AEGIS Weapon System	Development
	CV	Tactical Data System Aircraft Landing System Missile Fire Control System	Deployed
Airborne Systems	E-2C	Tactical Data System	Deployed
	P-3C	Airborne Patrol System	Deployed
	S-3A	Airborne Weapon System	Production/ Deployed
	F-14	Avionics and Weapon De- livery System	Deployed
Undersea and Landbased Systems	Trident	Command and Control System	Production
	Pershing	Weapon System	Deployed
	SAM-D	Weapon System	Advanced Development

2. Computer System Architecture: The selection of computing equipments and their operating relationships, including the functions allocated to each computational unit;
3. Computer Program Architecture: The structure used in computer program design throughout the system, including allocation of functions to elements of the computer programs;
4. Software Definition, Design, and Implementation Methods: Techniques used in software system design management and control, especially those which have had apparent success;
5. Software Validation and Integration Methods: Management techniques, testing tools and techniques, and facilities used in software quality assurance;
6. Software Acquisition Management Organization and Methods: Methods used by the Government, system contractor, and software contractor to manage the process of software design and validation; and
7. Operational Software Maintenance: Approach used or plans for transfer of developed software to Government control for life-time support and maintenance.

In the summary discussions that follow, a brief description, history, and highlights of each system are presented, noting plans for growth as well as current status. The Highlights are aspects of the system development that relate to the conclusions of this study. Each Highlight is annotated with the most closely associated study recommendations.

## 4.2 SHIPBORNE SYSTEMS

The current and projected Fleet deployment of ships either carrying digital equipment or planned for digital systems, in whole or in part, includes 12 carriers, 29 destroyers of the DDG class, 7 cruisers of the nuclear-powered DLGN-36 and DLGN-38 classes (to be redesignated), 30 cruisers of the DLG class (to be redesignated), and a unique cruiser (CGN-9).

Four of these ship classes, together with the Navy's newest air defense combat system (AEGIS), were selected for study since they represent successive phases in the evolution of digital Weapon Systems in the U.S. Navy.

USS Wainwright (DLG-28) exemplifies the currently deployed Naval Tactical Data System (NTDS) with the recent addition of digital fire control capability.

USS Towers (DDG-9) represents recent updating of selected Guided Missile Destroyers with a digital Tactical Data System.

USS Virginia (DLGN-38) is one of a new class of Guided Missile nuclear frigates now under development and using an extensive centralized complex of modular AN/UYK-7 computers.

USS Nimitz (CVAN-68) is the most recently commissioned aircraft carrier with digital Tactical Data and Aircraft Control Systems. Digital missile fire control systems are also installed on certain ships of the CV class.

A Strike Cruiser was selected to represent a possible future ship outfitted with the AEGIS Weapon System.

Figure 4-1 represents the relative complexity of digital systems in the selected ships. The figure shows major computers, peripheral memory, and operator consoles. These are segmented by the basic functions of a combat system: detection, combat direction, weapon control, and operability testing. The memory available to system computers is represented by the vertical scale; processors are represented by symbols. The number of operator consoles involved in these functions is also represented by symbolic blocks. The figure shows the growing trend toward automation, particularly in the functions of detection and systems test. This trend is accompanied by a reduction in the number of operators required for manual tasks of target position plotting.

Table 4-2 summarizes the types of digital computers employed in these shipboard systems. There has been a strong tendency to standardize the computer equipment used in shipboard combat systems. The USQ-20, CP-789, and CP-848 computers have been used in a variety of ship applications. The AN/UYK-7 and AN/UYK-20 computers implement more recent technology and are designated by the Navy as "standards" for current and future shipboard tactical digital applications.

USS Wainwright (DLG-28), deployed in 1966, was the first operational ship to carry a complete NTDS. The growth of digital capability in the Fleet from the initial digital systems in support of command and control to the present inclusion of digital fire control has been an evolutionary process, resulting from early pioneering in this area by the Naval ship Engineering Center (NAVSEC).

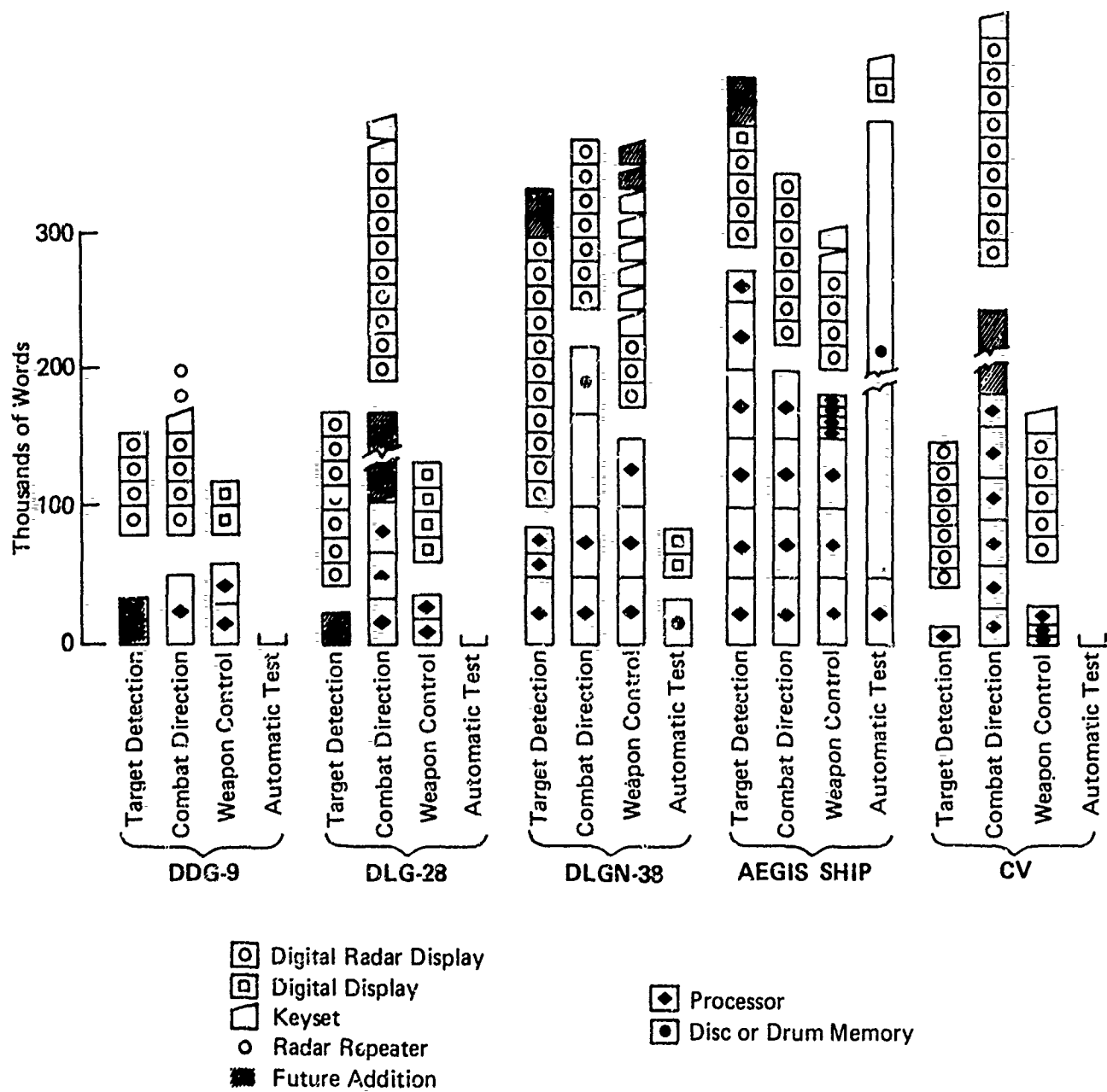


Fig. 4-1 Comparison of Shipborne Systems

TABLE 4-2  
SHIPBORNE COMPUTERS

Computer Designation	Word Length (bits)	Cycle Time ( $\mu$ s)	System	No. CPU's
CP-642/USQ-20	30	8	DLG-28	3
CP-642B/USQ-20	30	4	CV	5
CP-789/UYK	18	4	DLG-28	2
			CV	2
			DLG-28	2
CP-848/UYK	18	2	DDG-9	2
			CV	2
			CV	1 (option)
Mk 157	16	1	DDG-9	1
AN/UYK-7	32	1.5	DLGN-38	6
			AEGIS Ship	13
			DLG-28	3 (future)
AN/UYK-20	16	0.75	DDG-9	2 (future)
			AEGIS Ship	5

Responsibility for acquisition and maintenance of software and systems has been divided among several participating organizations with equipment responsibility typically at NAVSEC and software responsibility at FCDSSA, Dam Neck or San Diego. A significant part of the testing of these systems is conducted at the Navy's Mare Island Facility and aboard Fleet units.

Both digital hardware and software have advanced through many stages of development. The lessons learned within the framework of the NTDS development have led to several generations of standard computers, of which the AN/UYK-7 and AN/UYK-20 represent the latest. In a similar manner, standard displays have also evolved with the AN/UYA-4 Display Group being the latest example. Methods of programming have advanced from the dedicated processing of the early years to substantial amounts of shared memory processing and some use of multiprocessing. Recent hardware developments have shown a growing interest in the use of limited special purpose processing (e.g., microprocessors). Lessons learned in these programs have indicated a growing need for:

1. Strong program management, from inception to deployment maintenance,
2. Detailed documentation requirements,
3. Standard high level language development (CS-1, CMS-2),
4. Standard equipments, general and special purpose,

5. Functional system segments and common modules,
6. Firmly controlled interface specifications, and
7. Rigorous control of testing and changes.

The trend in shipboard digital instrumentation has progressed to include digital fire control in a significant number of our surface combatants. The development of these fire control systems has been independent but has made extensive use of NTDS equipments and technology.

Implementation of automatic detection and track (ADT) of sensor information has received substantially less attention than is desirable. A requirement for IFF Beacon Video Processing (BVP) was imposed on the Fleet during deployment in SEASIA to cope with the heavy track load and maintain appropriate classification of targets. A number of other systems are currently undergoing development, selected elements of which will be installed in future ship improvement programs. Among these are the AN/SPS-48C radar and the AN/SYS-1 integrated automatic detection and track system.

Major support systems aboard ships are also undergoing rapid change with automatic digital processing and control providing the prime forcing factor. Among these are systems for navigation, communication, and specialized functions such as aircraft landing control systems. A major growth in force and Fleet level coordination of command, control, and communications (C<sup>3</sup>) functions is expected in future years.

Increasing use of computers is being made to control system operational readiness test and fault isolation. The Digital Daily Systems Operability Test System (DDSOT) on current ships has significantly enhanced Weapon System availability. The Operational Readiness Test System (ORTS) of AEGIS represents a major improvement in this area. Further development of computer controlled on-line testing is expected in future improvement programs.

Information was gathered on shipboard Weapon Systems through visits to Program Managers and contractors, as well as through reference to APL/JHU personnel who have been directly involved with associated programs. The principal agencies were visited and are listed in Table 4-3.

TABLE 4-3  
WEAPON SYSTEM PROGRAM VISITS

Weapon System Program	Agency Visited	Responsibility	Date (1975)
DLC-28	NAVSEA 6541 FCDSSA(DN)	Program Manager (Terrier) Development and Maintenance Agent	3/4 2/11-12, 3/12-13
DDC-9	NAVSEA 6542 FCDSSA(DH) Raytheon Co.	Program Manager Development and Maintenance Agent Software Contractor	3/4 3/13 3/17
DLCN-38	NAVSEA, PMS 378 NAVSEC 6172 FCDSSA(im) Raytheon Co.	Program Manager Development Agent Maintenance Agent Software Contractor	3/10 3/4 3/12-13 3/17
AEGIS	NAVSEA, PMS 403 RCA Corp.	Program Manager System Contractor	2/24 2/12-14
CV	FCDSSA(SD)	Development and Maintenance Agent	2/25-26

#### 4.2.1 DLG-28 Guided Missile Frigate

##### Description

The USS Wainwright (DLG-28) is one of 30 DLG's armed with the Terrier or Standard Missile (SM-1) missile Weapon System. DLG's 6-15 were recently modernized to the same functional capability as DLG's 28-35, as were DLG's 16-25. DLG's 16-25 have missile batteries fore and aft as contrasted to the DLG's 6-15 and DLG's 26-35 which have single batteries. These ships all have Naval Tactical Data Systems and Terrier Digital Fire Control Systems. The mission of the DLG is to operate independently or with strike forces, antisubmarine forces, or hunter/killer groups, to provide area AAW defense for convoys or amphibious assault forces, and to coordinate engagement of submarine, air, and surface threats.

The major subsystems of the DLG-28 combat system include sensors, the combat direction system and the missile, gun, electronic warfare, and antisubmarine Weapon Systems.

Figure 4-2 is a diagram of the DLG-28 combat system. Functional changes being provided by the introduction of the Standard Missile 2 (SM-2) capability to the DLG's 16-35 are indicated by shaded blocks.

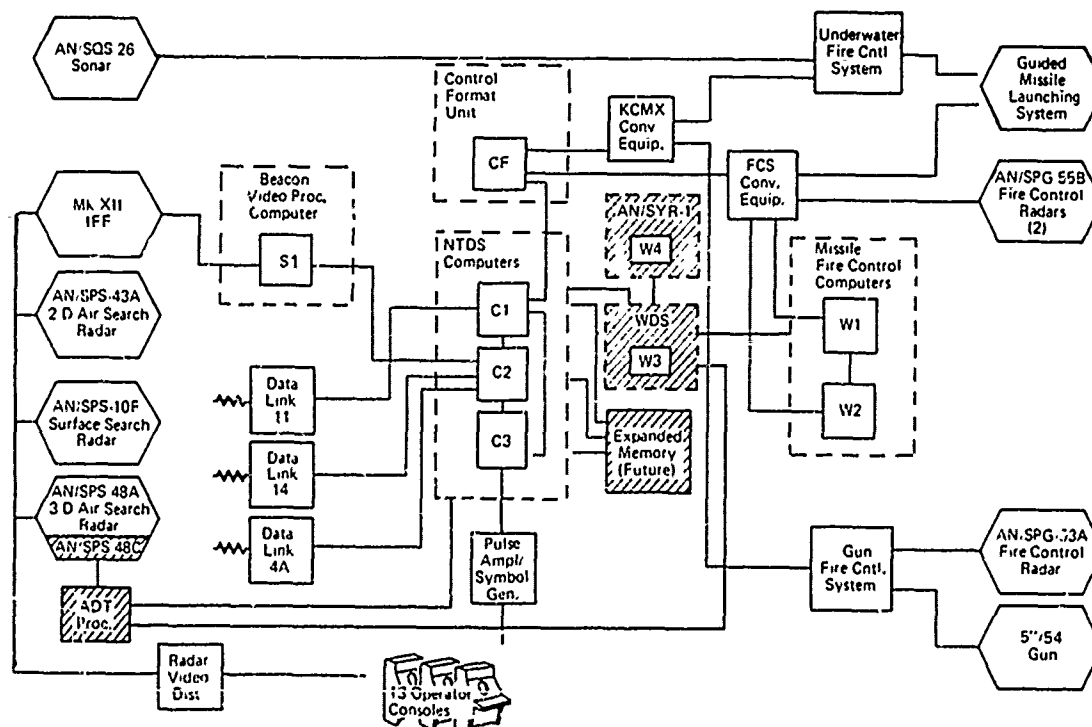


Fig. 4-2 DLG-28 Combat System



The DLG-28 was the first Navy ship with a fully automated Naval Tactical Data System (NTDS). The addition of digital fire control has further extended the digital capability. The future incorporation of an AN/SPS-48C radar with automatic detection and tracking (ADT) and an SM-2 missile capability will make this a highly capable ship. Operational functions and computer characteristics are given in Table 4-4.

TABLE 4-4  
COMPUTER SUMMARY

Unit	Type	Function	Processors	Memory
S1	CP-789/UYK (18 bit, 4 $\mu$ s)	Beacon Video Processing, Auto Detect and Correlate, Friendly Identification	1	16k
S2 (future)	AN/UYK-20 (16 bit, 750 ns)	Automatic Detection and Tracking	1	40k
CF	CP-789/UYK (18 bit, 4 $\mu$ s)	Control/Format Unit: Interface Control and Data Conversion	1	16k
C1, C2, C3	CP 642A/USQ-20A (30 bit, 8 $\mu$ s) or CP 642B/USQ-20B (30 bit, 4 $\mu$ s)	NTDS/WDS Mk 11: Rate-Aided Tracking  Threat Identification and Evaluation, Weapon Assignment and Control  Data Link Communication, Air Intercept Control	3	32k each
W1,W2	CP-848/UYK (18 bit, 2 $\mu$ s)	Missile Fire Control, Terrier Digital FCS Mk 76	2	32k each
W3 (future)	AN/UYK-20 (16 bit, 750 ns)	Weapon Direction System Mk 14, Weapon Assignment and Control	1	64k
W4 (future)	AN/UYK-20 (16 bit, 750 ns)	Communication Tracking Set AN/SYR-1 Processor, Control and Processing of SM-2 Missile Downlink Data	1	64k

### Acquisition History

NTDS Operational Program (Combat Direction System): The first operational NTDS for DLG's were deployed in 1962, and Service Test Programs, Model 0, were provided to several NTDS ships. These were limited capability programs designed primarily for support of anti-air warfare requirements. During the 13 years following this initial capability, NTDS programs have undergone extensive modifications both to structure and to operational capabilities. The present programs are designated as Model III programs. The most capable of these is the Phase 3 version, which supports ASMD improvements and the Digital Fire Control System on DLG-28 class ships. Beginning in July 1976, Model IV NTDS operational programs will replace Model III programs in DLG-28 class ships, as well as other units equipped with Link 11 communications equipments.

The 13 year period of development for DLG NTDS operational programs was accompanied by a learning process in the control and management of software development for complex real-time tactical data processing systems. Many of the procedures in present use resulted from lessons learned during early NTDS program development.

After the initial version of the NTDS DLG operational program was successfully deployed, it was turned over to the Fleet Combat Direction System Support Activity (FDCSSA) at Dam Neck, Virginia, for maintenance. Further modifications were then supervised by FDCSSA personnel. Since the initial program was turned over to FDCSSA, subsequent development has been almost entirely under the auspices of this activity. There has been no recent program acquisition as such, but there is a continuing process of modification by an on-site subcontractor under a level-of-effort contract. This is also true for in-process Model IV developments for DLG class ships. During the process of program revision, FDCSSA also acts as integration agent and validation agent for the Navy.

DFCS (Digital Fire Control System) Operational Program: Development of the DFCS Operational Program began in 1969. At that time the program was designated as the Terrier Adaptive Fire Control System (AFCS) computer program. APL/JHU designed the Advanced Development Model version of this program. During early design phases, the program was redesignated as the Digital Fire Control System computer program. Vitro/Automation Industries was assigned production responsibility. The first production DFCS was evaluated in 1972 aboard the DLG-26. A follow-on version of the DFCS was designated as the Universal DFCS computer program. The Universal Program was designed to operate with Mk 76 Mods 6, 7, and 8 DFCS's. The DLG-28 received DFCS modification in 1974.

### Management Information

	Naval Tactical Data System	Mk 76 FCS
Program Manager	FCDSSA(DN)	NAVSEA 6541 (Terrier)
System Contractor	FCDSSA(DN)	APL(ADM) Vitro (Production)
Software Contractor	ISSI	
Type Contract	LOE	
Program Status	Deployed	
Maintenance Agent	FCDSSA(DN)	NSWC Dahlgren
Software Deliverables	Operational Program and Program Documentation	Operational Program and Program Documentation
Validation Agent	FCDSSA(DN)	APL/NSWC (approved NSWSES, NAVSEA)
Integration Agent	FCDSSA(DN)	APL
In Service Engineering Agent		NSWSES

### Highlights

DLG-28 was the first operational (1966) combat system that integrated the tactical data system functions with weapon direction system functions resulting in WDS Mk 11. Weapon Systems requirements were identified by both the NTDS Program Manager and the Weapon Systems manager in mutually acceptable requirements documents. (MP1)

An on-site level-of-effort contract allowed close working relationships between the NTDS contractor and the development agent. Problems were easily identified, and required actions were taken earlier than would have been possible otherwise. (MP1, MS2)

The NTDS system can operate at reduced capacity with only two computers; the DFCS system can operate at reduced capacity with only one computer. (SE1, SE2)

The Dynamic Modular Replacement (DMR) technique for read-in of alternate program modules to facilitate different combat system warfare requirements was developed for these ships. (SE2)

An early freeze of Data Base design provided stable program development control. (SE3)

Common software modules were developed to provide compatibility among ships of the same class but with different equipment suites. (SE3)

The DFCS program was developed and validated using a land-based test site at the developer's site (APL). (IP3)

Both Navy and contractor personnel were experienced and knowledgeable in DLG combat system operational requirements. This enhanced development by exploiting proven development techniques and avoiding previous mistakes. (MS2)

A single agent was responsible for life-cycle maintenance as well as modification programming; this allowed an orderly transition from an existing program to a more capable program. (MS3)

#### 4.2.2 DDG-9 Guided Missile Destroyer

##### Description

The DDG-9 is one of 29 Guided Missile Destroyers armed with the Tartar or Standard Missile Type 1 (SM-1) (Medium Range) missile Weapon System. DDG's 2-24 were "new construction" while DDG's 31-36 are converted Sherman/Mitscher DD's. Four of the DDG's, C. F. Adams Class, are provided with a Tactical Data System. The mission of the DDG's is to operate with strike forces, with hunter/killer groups, in support of amphibious assault operations; and to screen support forces and convoys against submarine, air, and surface threats.

The major subsystems of the DDG-9 combat system include sensors, the combat direction and weapon control system and the missile, gun, electronic warfare, and anti-submarine Weapon Systems.

Figure 4-3 is a diagram of the DDG-9 combat system with the functional changes of the recently approved DDG upgrade program shown by shaded blocks. The upgrade is scheduled for only the DDG 2-24 (23 ships).

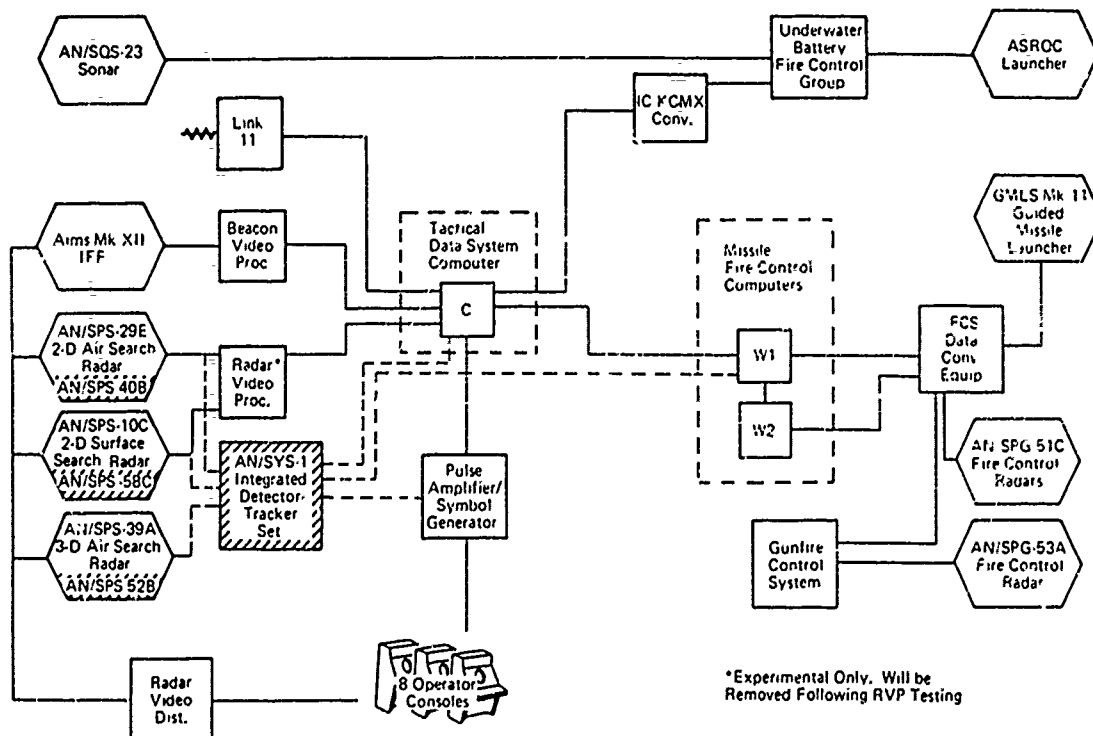


Fig. 4-3 DDG-9 Combat System

The DDG's carrying the Tactical Data System have also been improved by the addition of Digital Fire Control Systems. Additional improvements will include integrated automatic detection and tracking (IADT) by the addition of AN/SYS-1 IADT processors. A summary of functions and computer characteristics for the DDG class TDS ships is given in Table 4-5.

TABLE 4-5  
COMPUTER SUMMARY

Unit	Type	Function	Processors	Memory
C	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Tactical Data Processing; detect and track from beacon and 2D radar data, rate-aided track, threat identification and evaluation, weapon assignment, link communication	1	48k
W1	CP-848/UYK (Mk-152) (18 bit, 2 $\mu$ s)	Missile Fire Control/Weapon Direction	1	48k
W2	(CP-848/UYK (Mk-152) (18 bit, 2 $\mu$ s)	Missile Fire Control/Weapon Direction	1	48k

### Acquisition History

The development of Digital Fire Control Systems for the DDG's was initiated in 1964. The initial design was based on the use of the CP-789/UYK. As the development progressed, the capacity of the CP-789 was found to be inadequate, and the CP-848/UYK (Mk 152) was used. First installation was on DDG-16 in 1971, with all the DDG's to be completed by 1976.

The DDG Tactical Data System (TDS) program was initiated by NAVORD (Project Manager) in 1969. NAVSEC was designated as the Program Development Agency (PDA) with FCDSSA (Dam Neck) as the TDS agent, and Univac the prime contractor for software development. NSWC (Dahlgren) was selected as the NAVORD agent for the Tartar Weapon Direction System with Raytheon as the prime software contractor.

In 1970, software development was begun. Upon completion of an 18-month test phase at Mare Island (November 1971-1973), a performance test (DS 659) was conducted by OPTEVFOR (Operational Test and Evaluation Force) in mid-1973 at Mare Island. Final Formal Test and Acceptance by the Navy, conducted at FCDSSA(DN), culminated in a 24-hour endurance trial in early 1974. The Navy accepted the Operational Program (Version 0) on 1 April 1974. Version 2 has now been accepted. Three TDS DDG's (12, 15, and 21) have recently completed a successful 7-month tour in the Pacific. The TDS program is now in maintenance phase at FCDSSA(DN).

In the DDG-2 Class Upgrade Program, these ships will be the first to incorporate an Integrated Automatic Detection and Tracking (IADT) System, the AN/SYS-1. This system will initially provide the IADT function based on data from the AN/SPS-52B(Mod), AN/SPS-40C/D, and the AN/SPS-58C. Design goals for the future include the incorporation of all onboard sensors into the IADT system.

The design of the combat system for the DDG-2 Upgrade will use standard general purpose consoles and computers throughout the system. The responsibility for the development and implementation of the complete combat system for the DDG Upgrade has been given to NAVSEA 6542.

### Management Information

	Tactical Data System	Fire Control System/ Weapon Direction System
Program Manager	NAVSEA 6542 (Tartar)	NAVSEA 6542 (Tartar)
System Contractor	Univac	Raytheon
Type Contract	Cost Plus Fixed Fee	Cost Plus Fixed Fee
Program Status	Deployed	Deployed
Maintenance Agent	FCDSSA(DN)	NSWC Dahlgren
Software Deliverables	Operational Program	Operational Program
Validation Agent	FCDSSA(DN)	NSWC Dahlgren
Integration Agent	NSWSES	NSWSES

Highlights

The FCS/WDS program was developed in accordance with WS-8506 (AP2)  
(Ref. 28).

The WDS Mk 13 Program was the first to incorporate an equipment scheduler that provides the FCSC with a recommended engagement schedule. If ordered (by the FCSC) to execute the schedule, the program controls the assignments of fire control radars to targets and the loading and assignment of the GML3 to the Fire Control Systems. (SE1)

For certain ships in the class, the program will provide solutions and control for the simultaneous engagement of an air target with SM-1(MR) and the engagement of a surface target with SSSM(ARM). (SE1)

Prior to delivery to the ship, extensive system integration testing was conducted at the test facility at Mare Island. (IP3)

For the Tactical Data System, there was a single identifiable responsible agent for module design, coding, and implementation. (MS2)

For the Tactical Data System, the maintenance agent (FCDSSA) was involved throughout the program design, development, and integration phases. (MS3)

#### 4.2.3 DLGN-38 Guided Missile Frigate

##### Description

The USS Virginia (DLGN-38) is the first of five all-digital guided missile nuclear frigates armed with the Standard Missile (SM-1) (Medium Range) missile Weapon Systems. These ships will have missile batteries and light-weight 5"/54 gun systems fore and aft. The Combat System will utilize a central complex of AN/UYK-7 computers. The mission of the DLGN-38 is to operate with strike forces and to screen support forces and convoys against submarine, air, and surface threats.

The major subsystems of the DLGN-38 Combat System are the Combat Direction System, the sensor system (which is integrated into the Combat Direction System), and the missile, gun, electronic warfare, and antisubmarine Weapon Systems.

Figure 4-4 is a diagram of the principal elements of the DLGN-38 Combat System.

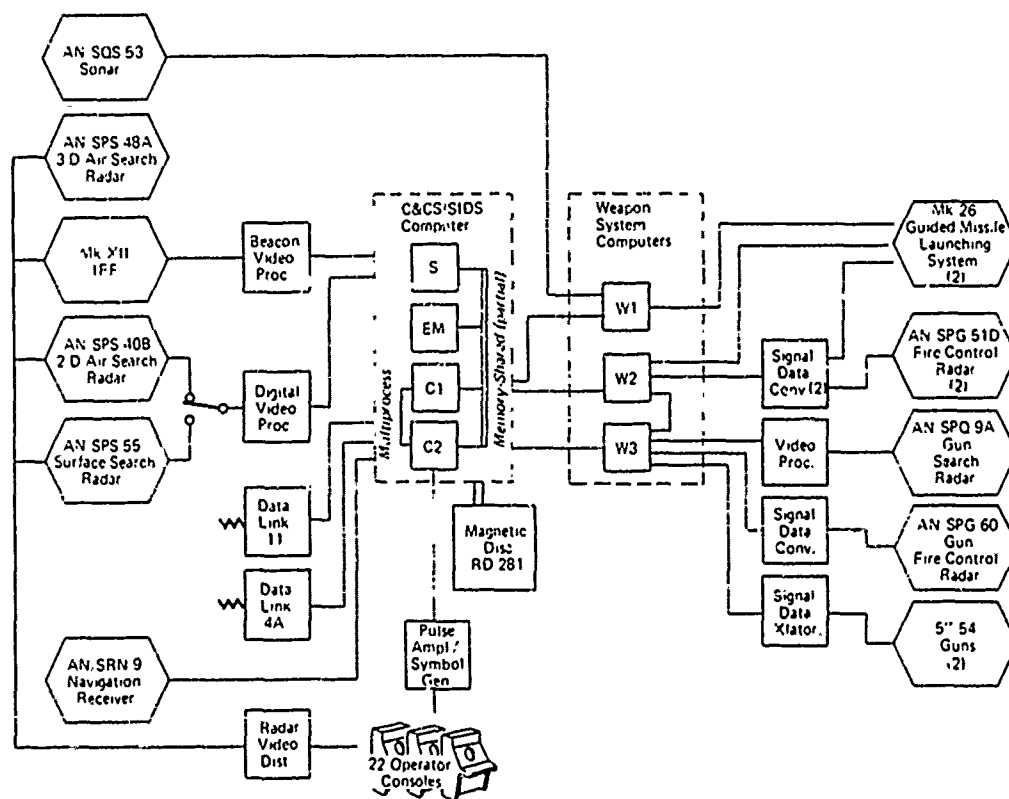


Fig. 4-4 DLGN-38 Combat System



The DLGN-38 represents the Navy's most advanced digitally automated combat system at the present time. The latest generation computers and displays are used throughout the system. Table 4-6 gives the functions and computer characteristics for this class ship.

TABLE 4-6

COMPUTER SUMMARY

Unit	Function	Processors	Memory
S	Sensor interface data system: Sensor data processing and correlation (BVP and DVP)	1 CPU 1 IOC	48k
EM	Extended memory, interface between SIDS and C&CS computers	1 IOC	64k
C1	Command & Control (multiprocessing): Target tracking, evaluation, weapon assignment, display, navigation, data link, ASW support, air control, surface operations, test control, training	1 CPU 1 IOC	48k
C2		1 CPU 1 IOC	48k
W1	Underwater fire control (Mk 116)	1 CPU 1 IOC	48k
W2	Missile fire control (Mk 74)	1 CPU 1 IOC	48k
W3	Gun fire control (Mk 86)	1 CPU 1 IOC	48k

Note: All computers listed are AN/UYK-7 units, with 32 bit word length, 1.5  $\mu$ s basic cycle time.

### Acquisition History

The DLGN-38 commenced Concept Formulation/Contract Definition in February 1968. The specifications for building Nuclear Guided Missile Frigate DLGN-38 and contract drawings were approved on 21 November 1969. During the same period, the DLGN-38 Ship Acquisition Project Manager (SHAPM) developed plans for the overall ship procurement, financial management, configuration control, procurement of Government-Furnished Equipment/Government-Furnished Information, and other items as set forth in a Ship Acquisition Plan.

Computer Program development has been structured in three phases: Program Production and Initial Integration, Program Integration and Shipboard Delivery, and Program Maintenance. The first phase, which is not yet completed, includes requirements determination, specification, program production, and testing at the Shore System Integration Site. The second phase will include shipboard testing up to and including Operation Program Functional Checkout (OPFCO). The third phase will involve completion of Final Contract Trials and ship turnover to the operational command for life-cycle maintenance.

### Software Management Information, DLGN-38 Class

	C&CS/SIDS	Mk 116 UFCS	MK 74 FCS	Mk 86 FCS
Program Manager	NAVSEC	NAVSEA	NAVSEA	NAVSEA
System Contractor	Univac	NUC	Raytheon	Lockheed
Type Contract	CPFF	WR	FP&LOE	LOE
Program Status	Ship Testing	Ship Testing	Ship Testing	Ship Testing
Maintenance Agent	FCDSSA	NUC	NSWC/ Dahlgren	NSWC/ Dahlgren
Software Deliverables	Operational Program and Program Documentation			
Validation Agent	NAVSEC	NAVSEA	NAVSEA*	NAVSEA
Integration Agent	Combat System Integration Manager			

\*FCDSSA assigned as SHAPM agent to witness and report to SHAPM on program certification.

### Highlights

DLGN-38 development requirements were primarily derived from those of the immediately preceding frigate class (which was itself still in development). (MP1)

The DLGN-38 Command and Control System (C&CS) has significantly overrun allocated computer resources. This can be attributed to changing and growing requirements during system development and errors in contractor estimates of module size. (MP1, SE2)

The Integrated Combat System Management Plan (ICSMP) (Ref. 29) is derived from the DLGN-38 Ship Acquisition Plan. The ICSMP provides for management of combat system integration by establishing and coordinating the schedules, work plans, and facility requirements of agencies participating in development. The tasks of this document provide guidance for the overall Combat System Test Plan (CSTP). (AP1)

The DLGN-38 Sensor Interface Data System (SIDS) is the first Navy attempt at an automated integrated sensor system. The SIDS/C&CS computer complex is also a pioneering example of separate software system developments within a shared-memory and multiprocessing architecture. (SE1)

Standards in force at the time of Contract Definition were applied to the selection of computer type, peripheral and switching equipment, consoles, language, and documentation. (SE1)

Software control is vested in the Software Configuration Control Board under the direction of the Fleet Combat Direction Systems Support Activity (FCDSSA(DN)). This board exercises control through a series of reviews and audits. Its establishment has resulted in a single agency (FCDSSA) being assigned the responsibility for implementation of design audits, Interface Design Specifications (IDS), and software Engineering Change Proposals (ECP). (SE1, MS3)

Extensive verification activity has been carried out at the Mare Island Test Site to determine that the Interface Design Specifications (IDS) had been correctly implemented. A separate facility is being assembled at FCDSSA(DN) for operational support and operator training. Simulation programs for subprogram development were funded as a recognized element of the program. (IP1, MS3)

Management and configuration control is facilitated by establishing baselines. These baselines serve as technical references from which the system elements will evolve to become operational systems. Since MIL-STD-480 (Ref. 30) and NAVMATINST 4130.1A (Ref. 18) do not detail the configuration control of computer programs to the degree necessary for DLGN-38 Class Combat Systems, the Software Configuration Control Procedures Manual (Ref. 31) expands basic terms and adds new terms where necessary. (AM2)

#### 4.2.4 AEGIS Weapon System

##### Description

The AEGIS Weapon System is a fast-reaction, high performance Weapon System that is being engineered to fit a wide variety of ship platforms. It is designed to provide the Fleet with a wide area surface-to-air and surface-to-surface defense through the 1980's and beyond by countering aircraft, antiship missiles, and launching platforms. When used with long-range surface-to-surface cruise missiles and extended range surface-to-air missiles the system will provide the Navy with a major offensive surface strike capability.

AEGIS is an integrated Weapon System consisting of a multifunction phase-phase array, fire control and weapon control systems, missile and launching systems, and a command and decision system. Major elements of a typical AEGIS-equipped cruiser are shown in Fig. 4-5.

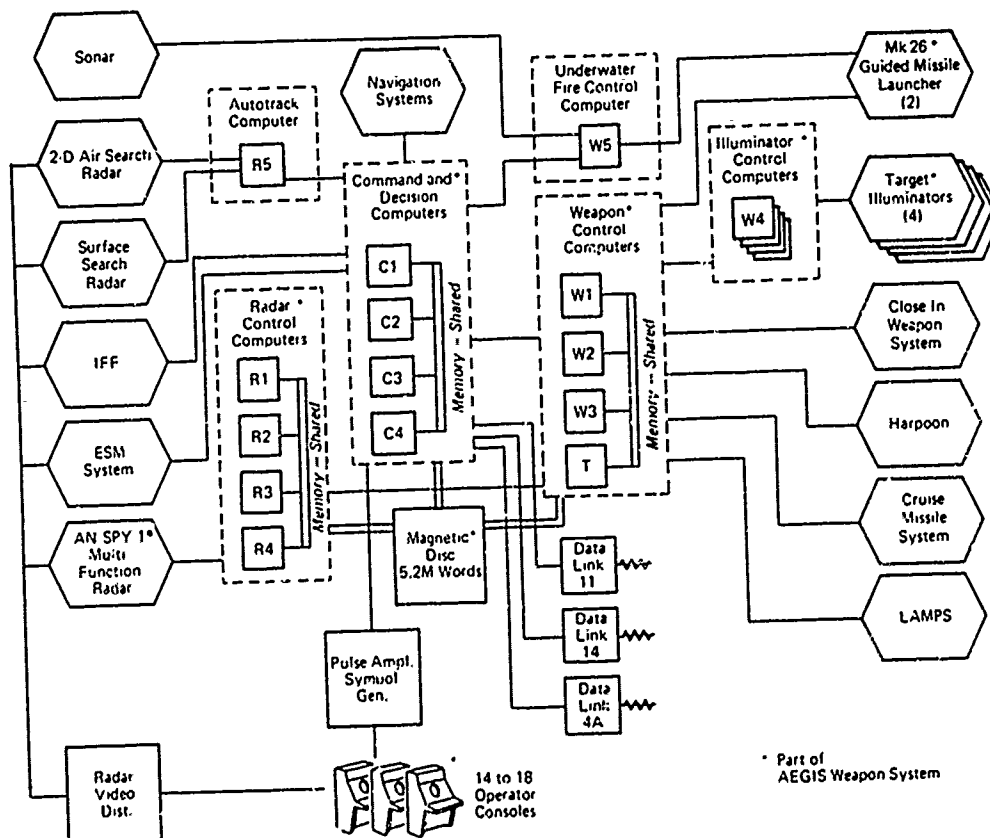


Fig. 4-5 AEGIS: Projected Ship System

The AEGIS system is fully automated, includes the latest model Naval Tactical Data System (NTDS), and makes use of new generation NTDS computers (AN/UYK-7 and AN/UYK-20) and displays (AN/UYA-4). Principal functions of the system and the resulting impact on computer characteristics are shown in Table 4-7.

AEGIS was originally engineered for installation on the DLGN-38 frigate class (now re-designated CGN-38 class). Simplification studies led to a modular system design which is adaptable to a variety of hulls. Current planning is directed toward installation aboard new construction Strike Cruiser and Destroyer class ships. Modernization of older cruisers with AEGIS is also under consideration.

TABLE 4-7  
COMPUTER SUMMARY

Unit	Type	Function	Processors	Memory
R1, R2, R3, R4	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Radar control, search scanning, automatic target detection and track, communication with guided missiles	4	256k
C1, C2, C3, C4	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Multisensor data correlation and management, identification, threat evaluation and weapon assignment, air intercept control, display, communications, surface operations	4	256k
W1, W2, W3	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Weapon direction and fire control, ASW support, LAMPS support	3	256k
T	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Operability test control, fault isolation	1	-
W4	AN/UYK-20* (16 bit, 750 ns)	Illuminator control	4	64k each
R5	AN/UYK-20* (16 bit, 750 ns)	Auto detection and track from beacon and 2D digitized video data	1	32k
W5	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Underwater battery fire control	1	48k

\*Under consideration

### Acquisition History

Requirements and conceptual definition for an Advanced Surface Missile System were derived during an intensive Navy/industry study conducted in 1963 under the direction of Rear Admiral F. S. Withington (Ret.). APL/JHU assisted the Navy in the conceptual design of the AEGIS system, addressing special effort to the more difficult technical requirements. Design, development, and demonstration of the essential elements of the multifunction array radar was accomplished at APL and provided a firm basis for a Full Scale Development of this new radar. In April 1968, DSARC I approval was given for initiation of Contract Definition. In 1969, following competitive proposal submissions by Boeing, General Dynamics, and RCA, a contract was awarded to RCA for the Engineering Development of the AEGIS Weapon System.

Program milestones A and B for Preliminary Design and Critical Design Reviews (PDR's and CDR's) were completed on schedule, and in October 1973 completion of land-based testing at the RCA plant signalled completion of milestone C.

During the initial engineering phase, program decisions were made to develop a functionally modular computer program and provide a Tactical Executive Program structured for AEGIS. The specifications and planning reflected strong emphasis on the software acquisition procurements.

Subsequent installation and tests at sea aboard USS Norton Sound have been successful and have generally supported program decisions made prior to and during the design reviews.

In June 1974 DSARC IIB endorsed the program and supported acquisition of the AEGIS system. Current acquisition planning is constrained by lack of an approved platform definition for the system. This problem is currently undergoing intensive study by the Navy.

### Software Management Information, AEGIS Weapon System

Program Status	Engineering Development (ED) (currently in Navy evaluation of first engineering model of two-phase ED program).
Program Manager	NAVSEA, PMS-403
System Contractor	(Prime) RCA Corporation
Type Contract	Cost plus fixed fee and incentive fee
Software Contracts	CSC, Raytheon, (RCA in-house)
Validation Agent	NSWSES
Maintenance Agent	TBD
Software Deliverables	Operational programs (3 systems) Compiler-monitor system Executive program (common) Software test and evaluation program Operational training program (5 systems) Operational readiness test program
Integration Agent	PMS-403

### Highlights

Design of the AEGIS system was based on extensive initial studies that established system requirements. Risk elements were largely removed through advanced development projects and competitive tradeoff analyses prior to contract award. The resulting system emphasizes modular, multifunction operation controlled by integrated UYK-7 computer bays using shared memory architecture and extensive on-line computer controlled operational testing and fault isolation. (MP1, SE1)

Required software deliverables in the AEGIS program require, in addition to the Tactical Operational Programs, a complete set of operational on-line and off-line test programs, and major support programs (Executive, Compiler/Operating System). (MP3)

PDR, CDR, and CAR (Configuration Audit Review) were scheduled at specific document delivery points. These were supplemented by frequent in-process reviews. (AP1)

The AEGIS program required a Computer Program Development Plan (Ref. 32) as a contract deliverable. This plan included development approach, work plan, schedule, and assigned responsibilities. (AP2)

Design control and auditing techniques employed included interface design documentation (AIID, CPID), Functional Flow Diagram & Description (F2D2), and program function tracing (Threads). (SE1, IP1)

The TADSTAND requirements for delivery reserve in computer resources are being applied, as well as additional growth reserves applied to current system sizing estimates. In addition to these, blocks of computer time and memory have been reserved for future support of systems now identified as future equipment for an AEGIS ship. (SE2)

Top-down design was employed in AEGIS EDM-1 and is planned for the total combat system. Standards and directives required in the design process included MIL-STD-490 (Ref. 8) amplified by WS-8506 (Ref. 28). SECNAVINST 3560.1 (Ref. 33) will be included in the next phase of acquisition. Documentation has been further extended to include an AEGIS Programmer Handbook setting forth rules and constraints for nomenclature and coding. (SE3, IP2)

A structured test plan included program unit (module) testing, testing of critical functional groups (Builds), segment testing with simulators, segment testing with equipment, and system integration tests. (SE3, IP3)

Major AEGIS facilities have included

1. Program Generation Centers for production and module-level program testing,
2. Factory Test Site for special-purpose equipment unit testing, including initial operation under computer control, where applicable,
3. Land-Based Test Facility for integration testing of computer programs with the total system, and
4. Shipboard Evaluation Facility for evaluation of system operation in an actual operating environment. (JP1, IP3)

Throughout the Engineering Development phase the Program Manager Office (PMO) has exercised strong control to ensure contractor compliance with contract provisions relating to software design and integration and test requirements with the hardware assemblies. The Government team for system review and contract monitoring has included computer system specialists in the program office, the NAVSEA Technical Representative's Office (at the contractor's site), the technical advisor (APL/JHU), FCDSSA(DN), and other contracted advisors. (MS1, MS2)

#### 4.2.5 Aircraft Carrier (CV) Tactical Data System

##### Description

The carrier forces planned for the U.S. Navy will consist of 12 ships divided into two fleets, the Atlantic and the Pacific. Four of the 12 carriers are or will be nuclear powered, giving them the advantage of being able to make long transits and remain on station for extended periods without refueling main ship propulsion.

Aircraft carriers can be assigned strike, support, or Antisubmarine Warfare (ASW) missions. The fighter-interceptor complement of the carrier air wing provides Fleet long range task group air defense capability. Effective adjuncts to the carrier's sensor system are embarked AEW aircraft with their HF data link capability. A vital element of fighter defense is the additional UHF data link between the ship, AEW aircraft, and interceptor aircraft.

Every carrier has two operational centers that deal with the command and control of aircraft: CIC and Carrier Air Traffic Control Center (CATCC). Each has its own supporting sensors. The CIC is concerned with the management of weapons for defense. The NTDS is an essential part of CIC operational support. CATCC is concerned with the safety of embarked aircraft. Major elements of a typical carrier Combat System are shown in Fig. 4-6.

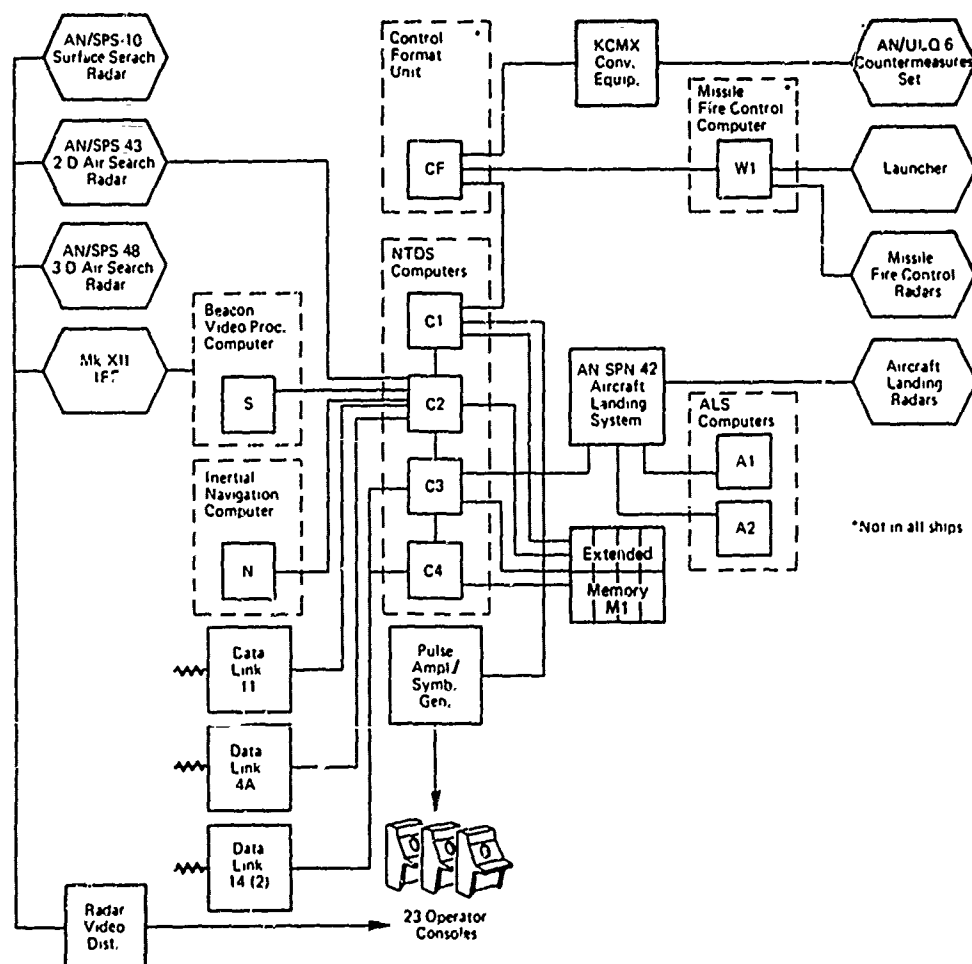


Fig. 4-6 CV Tactical Data System



The CIC and CATCC aboard the modern carrier are automated for high-capacity aircraft control. A listing of primary functions and computer allocations is given in Table 4-8.

TABLE 4-8  
COMPUTER SUMMARY

Unit	Type	Function	Processors	Memory
S	CP-789/UYK (18 bit, 4 $\mu$ s)	Beacon video processing: friendly identification, de- tection and track	1	16k
N	CP-642B/USQ-20 (30 bit, 4 $\mu$ s)	Navigation	1	32k
C1	CP-642B/USQ-20 (30 bit, 4 $\mu$ s)	NTDS: Rate-aided tracking, air traffic control, identification, threat evaluation and weapon assignment, display, communica- tions	1	32k
C2	CP-642B/USQ-20 (30 bit, 4 $\mu$ s)		1	32k
C3	CP-642B/USQ-20 (30 bit, 4 $\mu$ s)		1	32k
C4	CP-642B/USQ-20 (30 bit, 4 $\mu$ s)		1	32k
M1	MU-602(V)/UYK	Extended Core Memory Unit (ECMU) (642)	-	256k
CF	CP-789/UYK (18 bit, 4 $\mu$ s)	Control format unit: interface control and data conversion	1	16k
A1,A2	CP-848/UYK (18 bit, 2 $\mu$ s)	Precision tracking for aircraft landing control	2	16k (total)
W1	Mk 157 (16 bit, 1 $\mu$ s)	Missile fire control	1	16k

Future modifications to on-board carrier systems will include improvements in data link systems and command control communications systems.

### Acquisition History

The Fleet Combat Direction System Support Activity (FCDSSA) at San Diego has the responsibility for developing and maintaining NTDS software for this class of ship. FCDSSA(SD) designed and implemented the NTDS program for the CV, CG, DLG-28 class, and LCC. The development of the NTDS program for the CV used previously developed FCDSSA NTDS programs as a baseline design.

Each CV Operational Program is designated by a model and phase number. A model change involves a change in intership Link 11 message formats. A program may also be restructured when a model change is required. A phase change typically occurs every 2 years. It incorporates into the program library all minor changes that have accumulated in that time and that have been patched or deferred.

The NTDS program for the CV was developed at FCDSSA using in-house personnel to generate the Function Operational Design and level-of-effort contracting to generate the program design and code for each user module. The contractor worked at the FCDSSA facility which provided the necessary equipment and support software. All software design at FCDSSA(SD) is done by in-house personnel, and Navy software is used as the basis for all program development.

Prior to 1973 there were attack carriers (CVA's) and ASW carriers (CVS's). The difference in the tempo of operations made it impractical to mix the attack and ASW missions. Moreover, the ASW mission was not supported by the Naval Tactical Data System. With the introduction of the S-3 Viking aircraft, it was decided that any carrier should conduct any mission. Therefore, in 1975, carriers are deploying with both attack and ASW aircraft aboard. In 1974 the first F-14 squadron deployed on carriers. The data link associated with this aircraft required a small change in NTDS UHF link software to use feedback data from the aircraft.

### Software Management Information, CV Class Ships

Program Status	Deployed
Program Manager	FCDSSA(SD)
System Contractor	None
Software Contractor	Various plus in-house
Type of Contract	Level of Effort (tasks as necessary)
Maintenance Agent	FCDSSA(SD)
Validation Agent	FCDSSA(SD)
Integration Agent	FCDSSA(SD)
Software Deliverables	Phased program version and documentation deliveries

### Highlights

Software breadboarding was used extensively in the development of the F-14 capability in the CV program. FCDSSA(SD) recommends this in all new programs, to prevent re-design and slipped schedules. (MP1)

FCDSSA(SD) shows cost-effective results with its policy of in-house development using level-of-effort contracting. FCDSSA personnel cite programs acquired under end-item contracts (with the program code as a deliverable) which have resulted in costly programs that were difficult to maintain due to design or documentation incompatibility with their facility and procedures, since FCDSSA contracts for a software capability rather than a computer program. Separate funding is recommended for each task in the software acquisition process, with each task tied to a deliverable document. Funding of each task would be contingent on the review and acceptance of the previous task. (MP3, SE3)

In the development of the CV NTDS program, FCDSSA(SD) used a sequence of milestones through the analysis, design implementation, integration, test, and review process. Each milestone was associated with a specific deliverable document or program, and each was separately evaluated and an award fee paid accordingly. In-house capability is retained as a backup in the event of contractor failure, but since small components are contracted, no catastrophic failure can occur. (AP1)

FCDSSA(SD) uses support tools and facilities and periodically updates its support software. The most recent operating system, designated SHARE-7, operates in a UYK-7 computer and contains compilers, host-computer emulators, and debug aids. SHARE-7 is also used during the first phase of software validation and integration development at the program level. (IP1)

FCDSSA(SD) has an extensive test and integration facility which it shares with training. It provides capability for live mockup of systems and software under test. There are provisions for non-real-time and real-time operation, peripheral simulation, operator input or simulated operator input, and use of operational systems hardware. The FCDSSA(SD) Test Department provides test support, configuration control, delivery, and diagnostics. (IP3)

FCDSSA(SD) has acted as an agent for the NAVMAT Program Manager in some software acquisition. FCDSSA personnel cited their knowledge of operational requirements and believe it to be essential to developing an effective CDS program. They were able to make tradeoffs based on total life cycle costs because of their awareness of the total requirements. They strongly recommend having a user agent for the Program Manager involved throughout any Weapon Systems software acquisition process. (MS2)

Since FCDSSA(SD) is both the developer and the operational support agent, there is no transfer or duplication of either support software or of test and integration facilities. No cost or coordination is required. (MS3)

#### 4.3 AIRBORNE SYSTEMS

The four airborne systems selected for study represent several evolutionary lines of system development that provide the Fleet with Airborne Early Warning (AEW), Antisubmarine Warfare (ASW) capability and fighter-interceptor capability.

The E-2C Tactical Data System evolved in early 1970 from the E-2B and was deployed in 1974. It incorporated a new radar, display system, and passive detection system. The first AEW aircraft of this series, the E-2A, employed a drum computer. The L-304F computer selected for E-2B and E-2C was the first airborne multiprocessor.

The P-3C and S-3A airborne systems are interrelated developments which stem from exploratory work at the Naval Air Development Center (NADC) in Warminster, Pennsylvania. These systems have automated the tasks of acoustic submarine detection and tracking, and assist in aircraft direction. Several computers have been used. The latest, the Univac 1832, is similar to the AN/UYK-7 computer used in shipboard applications.

The platform for the F-14 Avionics and Weapon Delivery System is the F-14 Tomcat, a carrier-based fighter-interceptor aircraft. The AN/AWG-9 weapon control system used provides a significant increase in weapons and surveillance capability over previous interceptors. Improved data links between this aircraft and surface units provide closer coordination of surface and airborne defensive actions. Separately developed computer programs are used in the weapon control system and in the avionics system. The Weapon System development started in the early 1960's and later became part of the F-14 aircraft. The first F-14 operational squadrons deployed in 1974.

Table 4-9 lists the computers employed in these four systems. There has been less tendency to standardize computer equipments for airborne systems than for shipborne systems. Current plans for an All-Application Digital Computer (AADC) may provide a modular family of computers that will meet the constraints of airborne systems.

TABLE 4-9

AIRBORNE COMPUTERS

Computer Designation	Word Length (bits)	Cycle Time ( $\mu$ s)	System	No. CPU's
OL-77/ASQ (Litton L-304F)	32	2.2	E-2C	2
CP901/ASQ-114(V)	30	2	P-3C	1
AYK-10	32	1.5	S-3A	2
CDC 5400B	24	1	F-14	1
Teledyne CP-1050	20	7.5	F-14	1

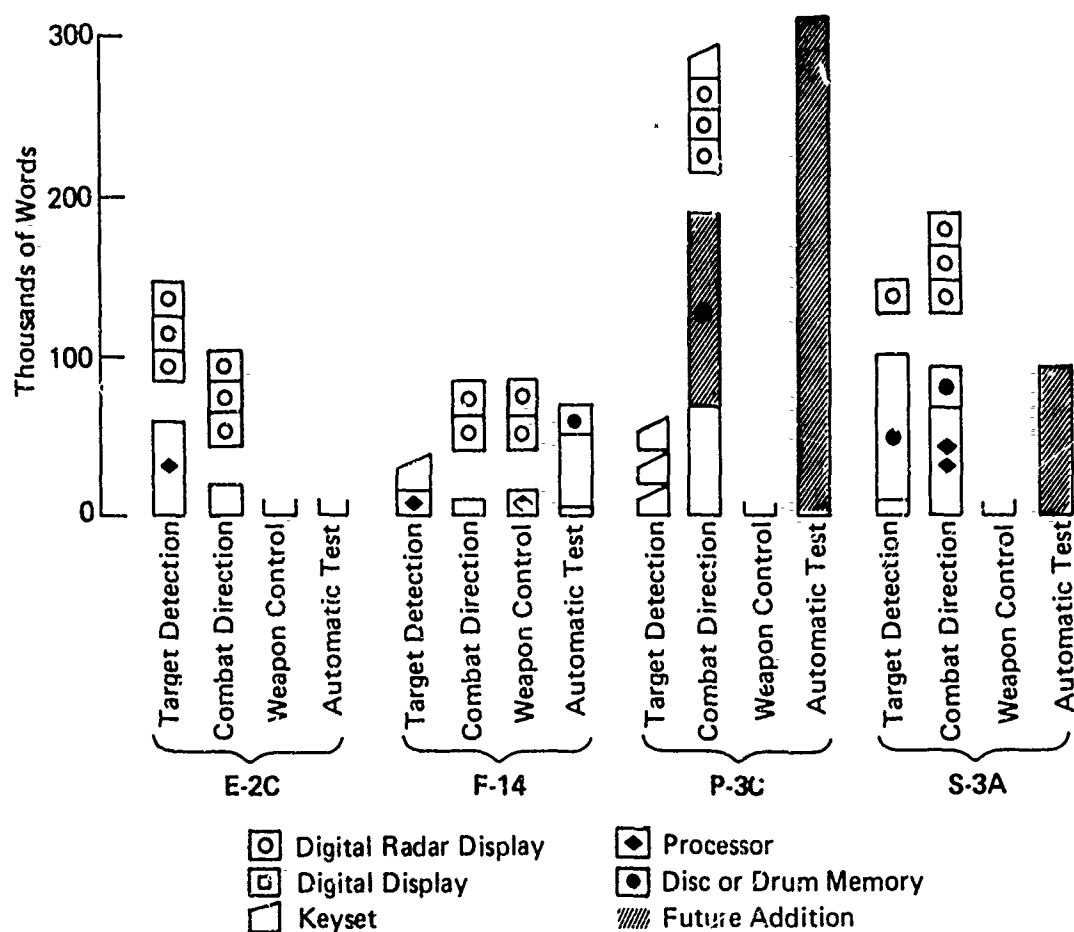


Fig. 4-7 Comparison of Airborne Systems

Figure 4-7 represents the computer and display capacities for these airborne systems, using the scale and notation that was used for shipborne systems in Fig. 4-1. Generally smaller than the shipboard systems because of size and mission limitations, these systems have expanded memory capability, where needed, by use of disk or drum peripheral units.

TABLE 4-10  
WEAPON SYSTEM PROGRAM VISITS

Weapon System Program	Agency Visited	Responsibility	Date (1975)
E-2C	NAVAIR PMA 231A NAVAIR 533 FCDSSA(SD) Grumman	Program Manager Comp. and Software Agent Maintenance Agent System Contractor	3/5 2/20 2/25-28 3/20
P-3C	NAVAIR 533 NADC	Comp. and Software Agent Advanced Development Agent, System Contractor System Contractor	2/12 - 3/5
S-3A	NAVAIR 533 NADC Lockheed	Comp. and Software Agent Advanced Development Agent System Contractor	2/12 - 3/5
F-14	NAVAIR PMA 241 VM NAVAIR 5331 NMC, Pt. Mugu Hughes Grumman, Pt. Mugu	Program Manager System Development Agent Maintenance Agent AWG-9 System Contractor F-14 Contractor	3/7 2/20 3/12 5/20-21 5/22

Table 4-10 lists visits made to program managers, support activities, and contractors in pursuit of information relating to these programs.

#### 4.3.1 E-2C Tactical Data System

##### Description

The E-2C is a carrier-based airborne tactical data and control system that provides radar early warning, passive detection, interceptor, and strike control capabilities.

The E-2C Airborne Early Warning (AEW) system uses three programmable computers. A dual-processor L-304F computer is the central processor. It performs the multisensor tracking and correlation, navigation and intercept vectoring, data link communication, and display generation functions. Special purpose computers are used in the passive detection and navigation systems. Figure 4-8 is a diagram of the E-2C tactical data system, and Table 4-11 gives the functions and computer characteristics for the E-2C system.

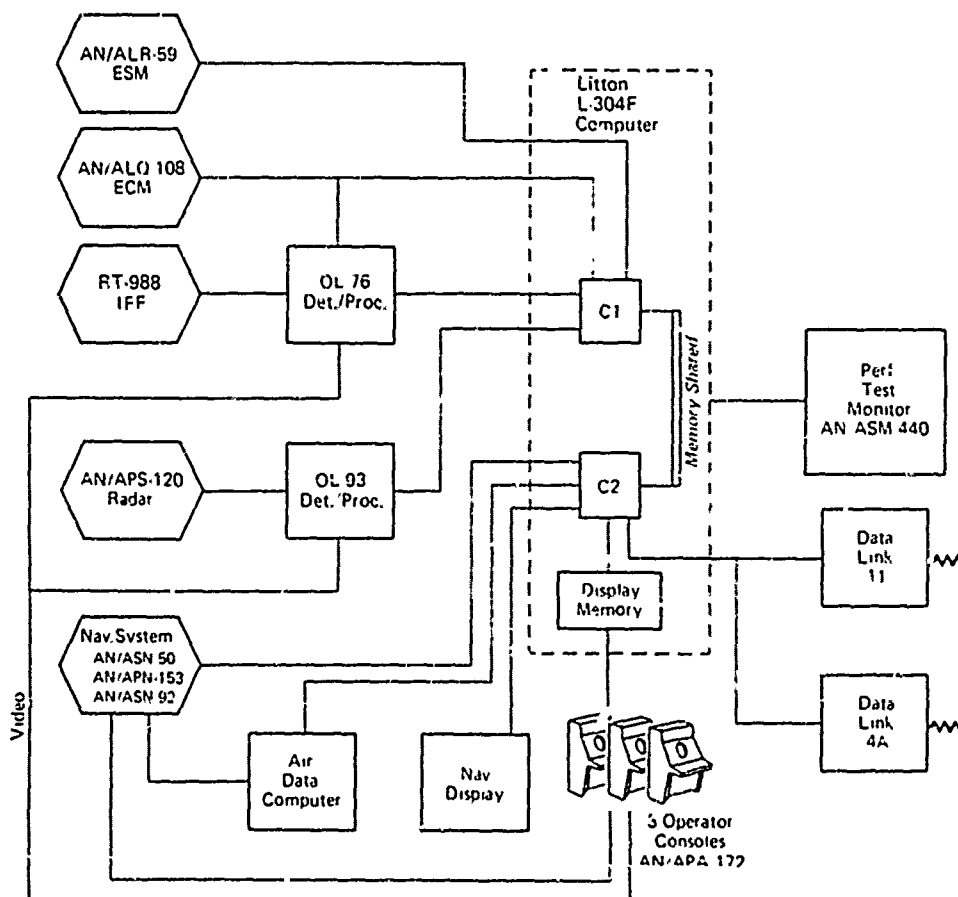


Fig. 4-8 E-2C Tactical Data System



TABLE 4-11  
COMPUTER SUMMARY

Unit	Type	Function	Processor	Memory
C1	OL-77/ASQ (Litton L304) (32 bit, 2.2 $\mu$ s)	Sensor processing and correlation, Link 11 control, test and monitoring	1	80k
C2	OL-77/ASQ (Litton L304) (32 bit, 2.2 $\mu$ s)	Navigation, display, intercept/strike control, Link 4 control, test and monitoring	1	

### Acquisition History

E-2C development from the E-2B began in early 1970. Primary changes included a new passive detection system, radar, and display system. The E-2C uses the same L304 computer as the E-2B. The Navy software maintenance organization, Fleet Combat Direction Systems Support Activity (FCDSSA), was a party to the software development phase, becoming involved in the early stages by assisting NAVAIR in writing the Functional Operational Specifications. Naval Air Test Center (NATC) and COMOPTEVFOR also provided support. A development contract (fixed price with incentive) was awarded to the system development contractor, Grumman Aerospace Corporation (GAC), with a separate line item for software development. Software design, generation, and validation were conducted by GAC using the software facility and system integration test site at their Bethpage facility. The software program was heavily documented. Direct FCDSSA involvement in the review and approval of the GAC test and evaluation plans proved highly beneficial to program reliability upon system deployment in September 1974. Transition of software from the contractor to Navy maintenance control was relatively smooth because of FCDSSA involvement in the development phase.

### Management Information, E-2C

Program Status	Deployed September 1974
Program Manager	NASC PMA-231, Capt. F. Roth
System Contractor	GAC
Software Contractor	GAC
Type of Contract	Fixed Price with Incentive
Validation Agent	GAC
Integration Agent	GAC
Maintenance Agent	FCDSSA(SD)
Software Deliverables	Operational Program, Functional and Design Description Documents

### Highlights

System growth requirements were recognized in the E-2C system definition with the result that a second L304 processor was provided for growth. This 100% margin was expected to more than absorb anticipated growth and computer load due to hardware uncertainties. However, largely because of increased radar data processing requirements, all of the margin was used, causing costly tailoring of the software to meet system requirements. (SE2)

GAC had a complete integration and test facility for the E-2C which contained all the actual hardware used in the system. It also had provisions for simulating many of the hardware interfaces. A similar facility was developed at FCDSSA(SD), the Operational Support Facility. The E-2C program had a very good reliability record, having logged 1200 hours with no software errors. GAC attributes a large part of this success to the extensive testing of the system. (IP3)

The E-2C Program Manager (PM) uses the NAVAIR Computer and Software Systems Group (Code 533) as technical support and review agent with success. The same agency is used by several NAVAIR PM's for this purpose. (MS2)

FCDSSA(SD) was tasked by NAVAIR under separate contract to provide support in the areas of E-2C program definition/specifications, program and document review, and test and evaluation planning. FCDSSA(SD) assisted NAVAIR in the preparation of requirements documents. (MS2, MS3)

Extensive Navy involvement in creating comprehensive test and evaluation plans contributed significantly to delivery of a program that has demonstrated excellent operational reliability. Navy involvement throughout the development phase also resulted in a smooth transition of software control from the contractor to the Navy support organization. (MS3)

#### 4.3.2 P-3C Airborne Patrol System

##### Description

The P-3C is a land-based Antisubmarine Warfare (ASW) patrol aircraft, with the mission to perform ocean surveillance, strike group and convoy protection, and mine laying operations. Detection, classification, and weapon delivery against surface and sub surface targets are basic requirements.

The airframe is a version of the Lockheed Electra. Electromagnetic, infrared, and acoustic sensors are used together with the visual capabilities of the crew. The aircraft system includes inertial, doppler, LORAN, and TACAN navigation units. The data processing system uses this and other tactical information to drive commands to a flight director system for use by the pilot.

Figure 4-9 is a diagram of the P-3C system and Table 4-12 gives the functions and computer characteristics for the system.

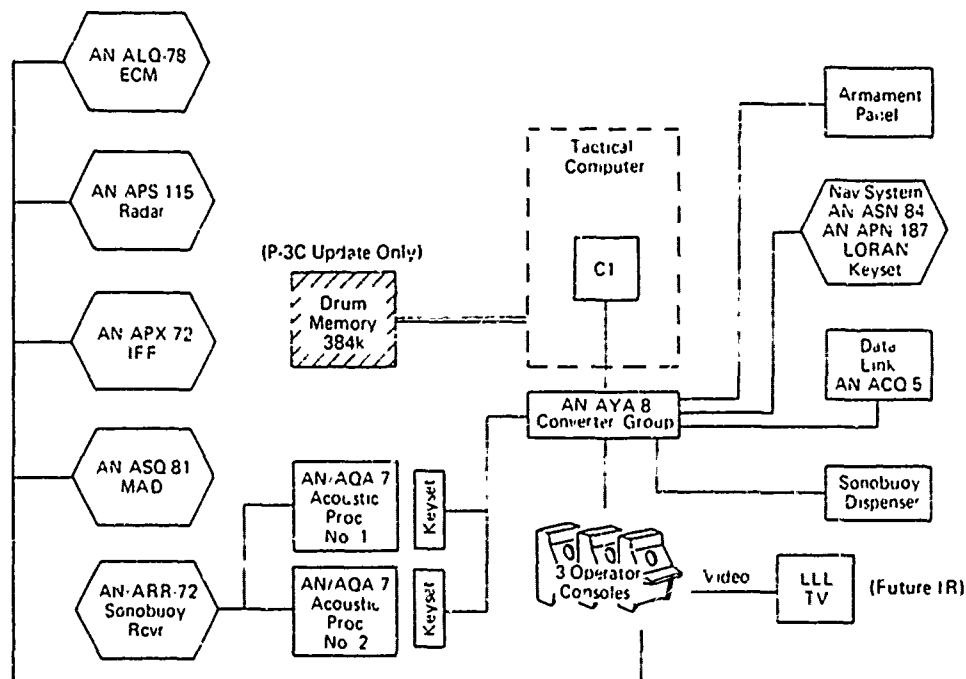


Fig. 4-9 P-3C Airborne Patrol System

TABLE 4-12

COMPUTER SUMMARY

Unit	Type	Function	Processor	Memory
C1	CP-901/ASQ-114 (30 bit, 2 $\mu$ s)	Navigation, storage of operator entered positions from sensors, sonobuoy tracking, stores inven- tory, stores auto drop control, display control	1	64k

### Acquisition History

Development started at the Naval Air Development Center (NADC) about 1960. It was originally pointed toward solving the S-2 tactical coordination problem and then shifted to the larger P-3 airframe.

A Mod 0 lab system was configured around a 32k, 30 bit CP-901 computer. There followed a Mod 1 flying configuration, a Mod 2 lab version, and finally a Mod 3 flying version that used an updated 64k memory of the CP-901.

The Navy began the P-3C program using the digital program in hand at NADC. In 1968 Lockheed received the NADC Functional Requirements Specifications (FRS).

A Software Management Team was established by PMA-240 who delegated control to NAVAIR 533. The remaining team members were Lockheed, Univac, General Electric, NADC, and Fleet Combat Direction Systems Support Activity, Dam Neck (FCDSSA(DN)). Periodic design reviews were held and design approaches were validated and demonstrated at the Integration Test Facility. VX-1 ultimately conducted an OPEVAL.

Upon delivery to the Fleet, FCDSSA(DN) took over maintenance support responsibility. Eight major versions of the program have since evolved.

Version A of the P-3C Operational Program was delivered by Lockheed in January 1969. At that time a Software Configuration Control Board (SCCB) was formed. In July 1969 version C was delivered. Version F, including ESM functions, was also delivered by Lockheed. FCDSSA(DN) has developed subsequent versions.

### Management Information, P-3C

Program Status	Deployed
Program Manager	PMA-240
System Contractor	Lockheed California Co.
Type Contract	Cost plus incentive fee (most equipment GFE)
Software Contractor	Univac
Validation Agent	VX-1
Maintenance Agent	FCDSSA(DN) (will be NADC in future)
Software Deliverables	Operational program, system test programs, diagnostics, functional requirements specifications, coding and design specifications, program listings
Integration Agent	Lockheed California Co.

### Highlights

The contract to Lockheed was originally for the equipment only, with the Navy supplying the computer program under NADC auspices. The contract was subsequently changed to include provision of the software by Lockheed. Lockheed's task then was to re-do the functions already demonstrated in the NADC prototype. About 20% of the production programmer team had previously been with the NADC prototype program. (MP1)

The GFE computer was chosen as a part of the NADC Mod 3 effort. It represented little or no risk since it had been part of the prototype development. The selected airborne computer was chosen as the constraining factor in program size. The CS-1 compiler was also GFE. (MP1, SE1, IP1)

The FRS were not in accordance with WS-8506 (Ref. 28) or similar standards. They were used to develop coding and design specifications. Program listings themselves provided the final form of documentation. The original Mod 3 NADC listings were used as references during the production software development. (MP3)

Based on the experience of the P-3C software development, the digital program developers indicate the need for a more structured programming approach to producing the final product. Along with this, a need was recognized for a "system level" document that would address hardware and software interactions and requirements in the same context. This would fill the gap between an operational level specification and the program functional requirements specification. (SE1)

The original computer used in P-3C had 32k words of memory. During development the computer was enlarged internally to 64k. The recent P-3C "Update" incorporated a 384k drum to perform expanded functional capabilities. System test programs comprise three to four times as much code as the operational program. (SE2)

Since the equipment was GFE, the documents describing its operations were not tailored for use by programmers. Lockheed instituted a Programmers Technical Manual (PTM) for each equipment of interest. These PTM's were written so that a programmer would understand the digital input-output interface reactions. (IP2)

A very critical part of the acquisition process was the development and availability of the Integration Test Facility which preceded coding. This early validation of code was considered critical. The test facility was made up of the major sensor, display, and computer interfacing equipment as well as the computer itself. Code was checked out in segments relating to individual subsystems and then brought together into a total system representation. (IP3)

The P-3C update is a major departure from the previous procurement of software. The Navy itself, using NADC, is designing and developing the program. The extensive Integration Test Laboratory at NADC has provided the same type of development facility as the one used by Lockheed. This step was taken by the Navy in order to achieve an alternative to sole source software procurement. (MS2)

### 4.3.3 S-3A Airborne Weapon System

#### Description

The S-3A is a carrier-based aircraft with the mission to perform ocean surveillance for convoy and strike group protection. Detection, classification, and weapon delivery against surface and subsurface targets are basic requirements.

The airframe is totally new. Two turbofan jet engines are specifically designed to meet dash and loiter speed requirements. Electromagnetic, infrared, and acoustic sensors are used together with the visual capabilities of the crew. The aircraft system includes inertial, doppler, LORAN, and TACAN navigation units. The data processing system uses these and other sources of tactical information to drive commands to a flight director system for use by the pilot. Figure 4-10 is a diagram of the S-3A system and Table 4-13 gives the functions and computer characteristics for the system.

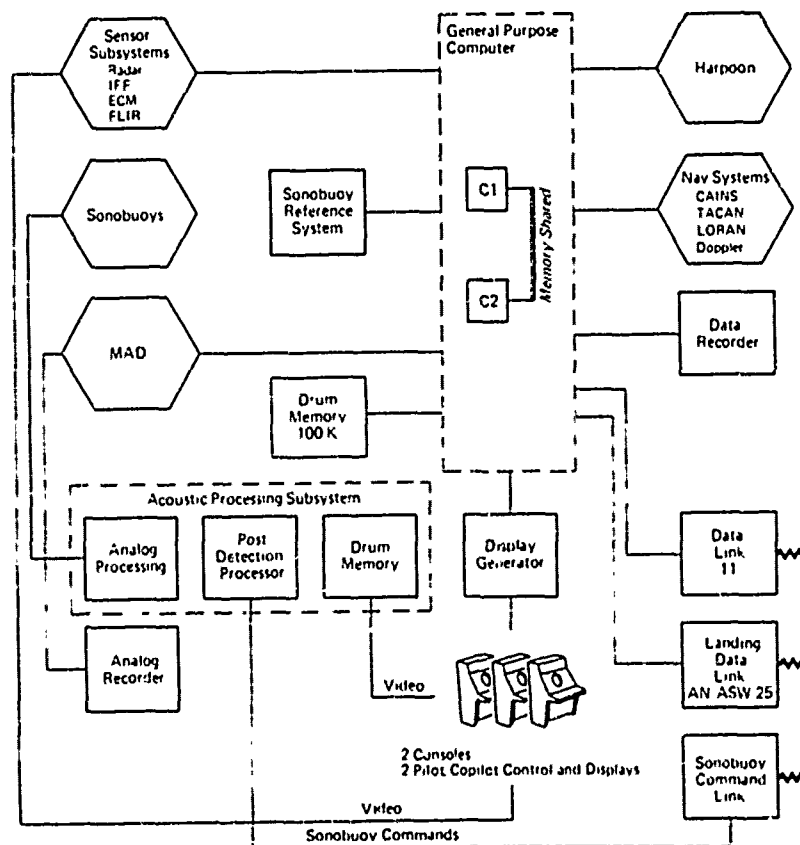


Fig. 4-10 S-3A Airborne Weapon System



TABLE 4-13  
COMPUTER SUMMARY

Unit	Type	Function	Processor	Memory
C1 & C2	AYK-10 (U 1832) (32 bit + 4 parity, 1.5 $\mu$ s)	Navigation, Harpoon launch control, target tracking, sonobuoy tracking and inventory, target classifi- cation, INCOS control, display control, system tests	2 (Multi- processed)	64k

#### Acquisition History

The award in 1969 of the S-3A contract to Lockheed was the culmination of the NADC ANEW program started in 1960. The original ANEW diversion to P-3C provided an experience base for the S-3A program. Lockheed subcontracted to Univac, its bid team member, for the software on a fixed price basis. An integration test facility was established at the outset for software/hardware interaction development.

Fleet Issue 1, the first Fleet Operational Program, was delivered in 1974. Fleet Issue 2, delivered in February 1975, is comprised of errata from the Board of Inspection and Survey (BIS). Fleet Issue 3 will include new data link and acoustic classification modifications. Ten operational squadrons will be outfitted with the new system by 1977.

#### Management Information, S-3A Aircraft Systems

Program Status	OPEVAL
Program Manager	PMA-244
System Contractor	Lockheed California Co.
Type Contract	Cost plus incentive fee (most equipment CFE)
Software Contractor	Univac
Validation Agent	VX-1
Maintenance Agent	NADC
Software Deliverables	Operational Program, system test programs, diagnostics, functional requirements specifications, coding and design specifications, program listings
Integration Agent	Lockheed California Co.

### Highlights

When Lockheed was awarded the S-3A contract, much of the system experience in both hardware and software was transferred from the P-3C efforts. The major increase in effort was in acoustic processing and classification and associated drum storage and display requirements.

The Performance Specification was written as a joint effort between the integration contractor (Lockheed) and the software contractor to ensure thorough mutual understanding. The contract for the software was fixed price. (MP1, MP3)

During 1969-1974 about 175 programmers generated 500 000 instructions. Roughly a third of the effort was used to generate the operational program. The rest was used to generate the system's test and diagnostics and the special development test software. (MP3)

The S-3A management of software included milestones listed for each primary "function." The programs were constructed with a building block concept that determined where milestones were logically sequenced. Standard Milestones and Weekly Progress Reviews were used for over 800 separate functions. (AP1)

A team concept was used in development, which required that the Design Engineer and Programmer work together daily and that the engineer understand programming language. (IP2)

A comprehensive integration and test support facility was developed for the S-3A development. Program checkout and a phased sequence of integration steps were accomplished using this facility. The facility, which used both actual and simulated equipment, minimized the need for flight tests to verify system performance. A flying test bed was required for final integration and testing. (IP3)

#### 4.3.4 F-14 Avionics and Weapon Delivery System

##### Description

The F-14 Tomcat is a high-performance carrier-based fighter interceptor. The primary missions of the total F-14/Phoenix System are Fleet Air Defense, Air Superiority (both Beachhead and Escort), Air Combat Maneuvers, and Interdiction. The F-14 Avionics and Weapon Delivery System has the capability of detecting and tracking high-altitude targets at long range, detecting high-altitude hot targets against a cool sky, maintaining 24 simultaneous sensor target tracks, maintaining 8 simultaneous data link tracks, looking-down for detecting and tracking low-altitude targets, engaging maneuvering targets in close-in "dogfights," engaging up to six separate targets simultaneously with the very-long-range AIM-54A (Phoenix) missiles, and using all other Navy air-to-air and air-to-ground weapons.

The major components of the F-14 system are the Sensor System, the Weapon System, the AWG-9 Weapon Control System, and the Computer Signal Data Converter (CSDC) Subsystem. Figure 4-11 is a diagram of the F-14 system and Table 4-14 gives the functions and the computer characteristics of the system.

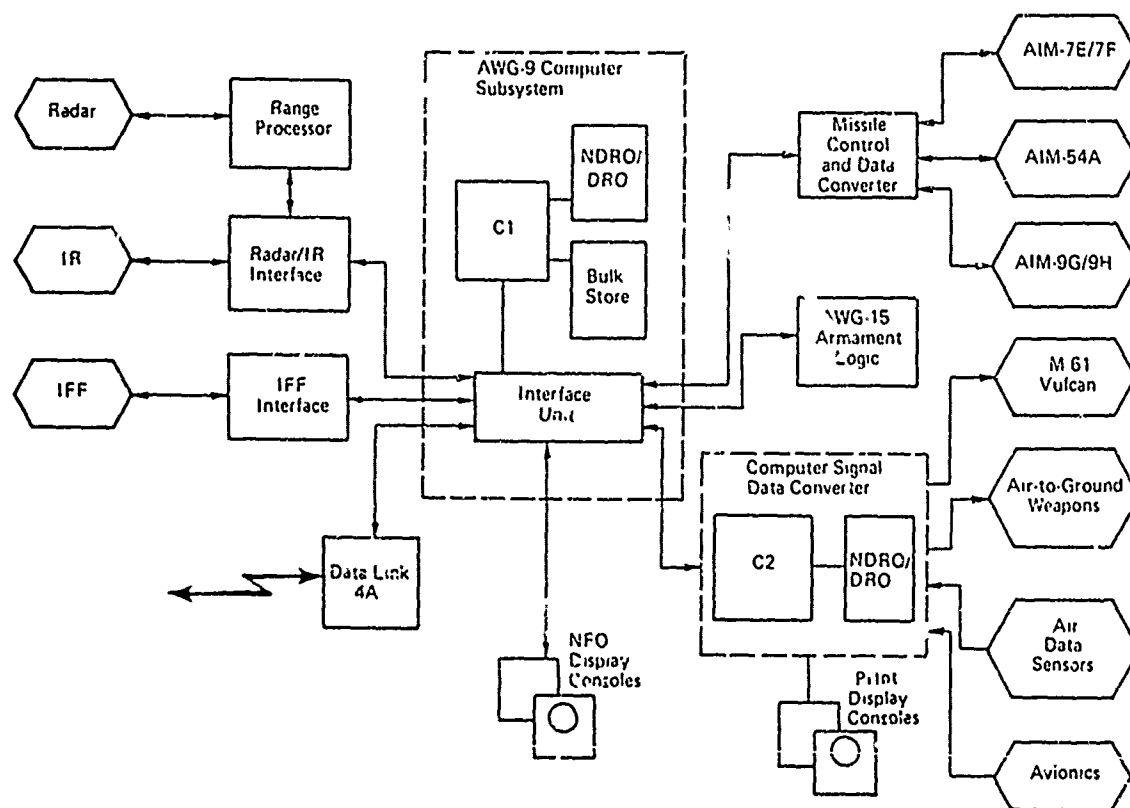


Fig. 4-11 F-14 Avionics and Weapon Delivery System

TABLE 4-14  
COMPUTER SUMMARY

Unit	Type	Function	Processor	Memory
C1	CDC 5400B (AWG-9) (24 bit, 1 $\mu$ s)	Target tracking, steering, display, missile launch zones and parameters, navigation, built-in-testing (BIT)	1	24k NDRO 8k DRO 140k Tape
C2	Teledyne Systems CP-1050 (20 bit, 7.5 $\mu$ s) (CSDC)	Platform management, coordinate transformations, avionics input/output, onboard-checkout (OBC)	1	1k NDRO 4k DRO

#### Acquisition History

The Navy announced that it had awarded a contract to Grumman in January 1969 for a new carrier-based fighter. Known as the VFX during the competition phase of the program, this aircraft was officially designated as the F-14 Tomcat.

First flight test of the F-14A prototype took place on 21 December 1970; 7 more F-14A's were flying before the end of 1971, and by early 1973 20 aircraft had logged almost 3000 hours in more than 1500 flights. Weapons System testing accounted for half of the total flight time.

The AWG-9/Phoenix concept was initiated in 1960 and Hughes Aircraft Co. was selected as prime contractor by the Navy in 1962. Flight testing began in 1965, and the first successful intercept was in September 1966. The simultaneous attack capability was demonstrated in March 1969 when two drones were engaged from an F-111B aircraft. Subsequent to cancellation of the F-111B, development of Phoenix has been in relation to the F-14 aircraft. F-14 flight trials started in April 1972, and in December 1972 four jet drone targets were successfully engaged by four Phoenix missiles launched and directed by the AWG-9 System of an F-14A Tomcat.

#### Management Information, F-14

Program Status	Deployed
Program Manager	PMA-241-VM
System Contractor	Grumman
Type Contract	Fixed cost
Software Contractor	Hughes (AWG-9), Grumman (CSDC)
Validation Agent	Grumman/Hughes
Maintenance Agent	SSA
Software Deliverables	Operational program tapes and documentation
Integration Agent	Grumman

### Highlights

An AWG-9 design approach based on expanded software interaction and control of system hardware functions permitted a rapid hardware development cycle. The system flexibility provided by the AWG-9 software has permitted AWG-9 system growth and functional improvement without hardware modification to in-service systems. (MP1, SE1, SE2)

Extensive flexibility and growth potential was designed into the AWG-9 computer interface hardware to permit integration of devices with a variety of interface types. A programmable input/output controller and interface capability for parallel, serial, DMA, and analog interfaces provides this capability. (MP1, SE1, SE2)

At the time that the AWG-9 was developed, Non-Destructive Read-Out (NDRO) memory provided the speed and security required for program instruction storage. However, other protection techniques now available would probably permit use of the more flexible DRO memories for new systems. (SE1)

The Metaplan compiler was developed for the AWG-9 to provide a machine-independent, high-level language with a stringent requirement for efficiency of generated code. Benchmark tests indicate an efficiency level of 90 to 95% has been achieved. (SE3, IP1)

The establishment of the System Integration Test Site (SITS) by Grumman permitted testing and validation of the system software prior to and during actual flight testing. Flexibility was thus obtained in the implementation of the system. The location of SITS at a government installation (PMR, Pt. Mugu) enhances the transferability of the system software to the Navy Software Support Activity (SSA). (IP3)

Validation and integration of computer programs is performed at simulation facilities at the Hughes Roofhouse facility (AWG-9) and at the Grumman SITS integration facility (total system level) at PMR. SITS also provided the facility for the first two-way Link 4A tests. (IP3)

#### 4.4 UNDERSEA AND LANDBASED SYSTEMS

One submarine system and two Army landbased systems were selected to complete the survey of representative software development programs.

The Trident submarine is in development as an upgrading of the current Fleet Ballistic Missile (FBM) fleet. Trident includes two major Weapon Systems, one strategic and one tactical. The tactical system, which was examined in this study, will employ the AN/UYK-7 computer which is now a standard for surface units.

The two Army systems examined, Pershing and SAM-D, have particularly stringent space, weight, and power requirements because of their need for mobility. Both systems have selected computers specially tailored to their needs. Pershing selected commercially available computers; SAM-D has developed a new computer.

The Pershing system has been deployed since 1964. SAM-D is in the advanced development phase.

Visits to agencies concerned with the development of these systems are listed in Table 4-15.



TABLE 4-15  
WEAPON SYSTEM PROGRAM VISITS

Weapon System Program	Agency Visited	Responsibility	Date (1975)
Trident	PM-2	Program Manager	2/11
	NUSC	Certification Agent	2/18
	EB/IBM	Eng. and Integ. Agent	3/7
	NAVSEC 6172	Project Director	3/12
Pershing	Missile Command, Redstone Arsenal	Program Manager	2/8
	Martin Marietta Aerospace	System Contractor	3/4
SAM-D	Missile Command, Redstone Arsenal	Program Manager	2/8

#### 4.4.1 Trident Command and Control System

##### Description

The primary mission of the Trident submarine is to host a strategic Weapon System capable of delivering Intercontinental Ballistic Missiles to selected targets and ensuring the invulnerability of that Weapon System by conducting undetected submerged patrols.

The Trident system offers significant advantages over the existing Fleet Ballistic Missile (FBM) Fleet in that it responds to:

1. The growth in Antisubmarine Warfare (ASW) capabilities by permitting operation in nearly four times the ocean area of the existing FBM Fleet because of a missile (C-4) range twice that of the Poseidon (C-3) equipped FBM Fleet;
2. The potential unavailability of overseas bases by permitting operation and home-porting out of continental United States bases; and
3. The age of some of the existing FBM Fleet. For example, the 598 Class SSBN's are approximately 15 years old.

The Trident submarine operational availability date is currently April 1979.

Although the Trident mission is strategic, the submarine is equipped with a tactical system for avoidance of encounters or, failing in that objective, for conducting a tactical engagement. The tactical Command and Control Subsystem was examined in this study. Figure 4-12 is a diagram of the Trident Command and Control System.

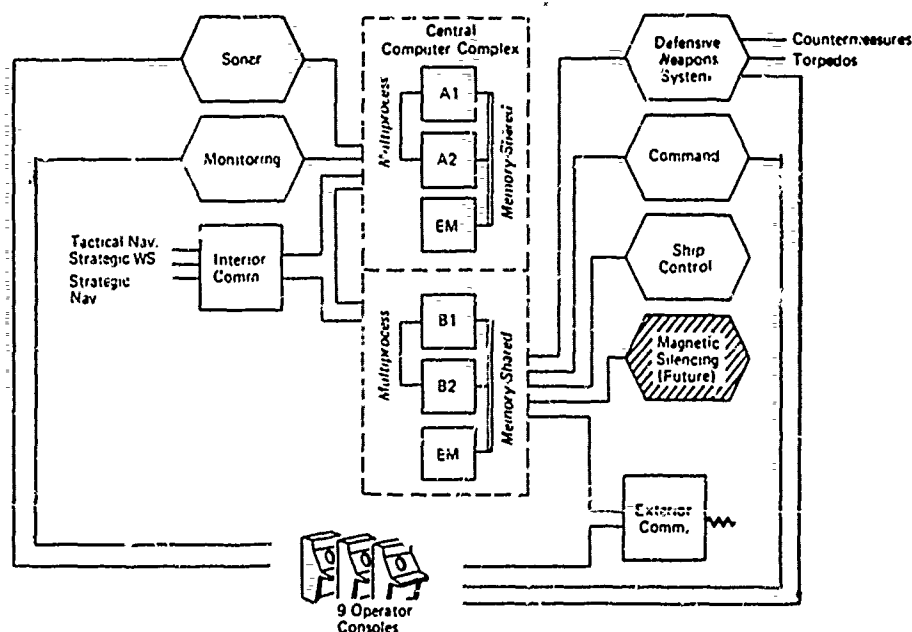


Fig. 4-12 Trident Command and Control System

Basic functions and computer characteristics for the Trident submarine are given in Table 4-16.

TABLE 4-16  
COMPUTER SUMMARY

Unit	Type	Function	Processor	Memory
A1, A2	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Sonar data processing, system monitoring	2	180k
B1, B2	AN/UYK-7 (32 bit, 1.5 $\mu$ s)	Support weapons systems, command, ship control, magnetic silencing	2	148k

### Acquisition History

Requirements for an integrated Command and Control System (CCS) were first formally specified in January 1972. A contract to Electric Boat (EB) to accomplish studies and tradeoffs leading to a proposed configuration was awarded in early 1972. EB subcontracted to IBM Federal Systems Division to develop studies leading to a Proposed Technical Approach (PTA). The PTA was delivered in the fall of 1972. It was subsequently revised, condensed, and reissued as a Navy document in early 1973.

A system level specification and a design data document were issued in the summer of 1973, which formalized the system configuration.

Level I interface testing was conducted from August to December 1974. Level II (partial program) testing commenced in January 1975 and is currently in progress.

### Management Information, Trident

Program Status	Production
Program Manager	PM-02
System Contractor/Integrator	EB/IBM
Type Contract (System and Integrator)	Cost plus fixed fee
Software Contractors	
Sonar	IBM
Defensive Weapons/Command	NUSC
Ship Control	EB
Common/Service Programs	Univac
Maintenance Agent	Trident Maintenance Agency
Software Deliverables	As per WS-8506 (Ref. 28); plus Operating procedures Source and object library Memory map Compiler and loader card decks Compile listing tape
Certification Agent	NUSC

### Highlights

Trident treats operational software and Test and Integration software as configuration items. The compiler, CMS-2Y, was furnished by the Government and not called out as a deliverable. (MP3)

The Trident Software Management Plan (Ref. 34) defined five development phases with specific deliverables. The development phases were: planning, program generation, integration test and evaluation, shipboard installation and acceptance, and program maintenance. Deliverables included programs, documentation, plans, and test facilities. (AP1)

Subsequent to the Proposed Technical Approach (PTA) study, early software management planning defined schedules, organization, standards and conventions, implementation methods, operating philosophy, integration strategy, resource control/allocation, and the Land-Based Evaluation Facility. (AP1, AP2)

Specified design and implementation techniques such as top-down design and structured programming allowed defined and phased delivery of software components, thus permitting early interface/integration testing. (AP1, SE3, IP2)

Prior to DSARC II, Trident conducted an in-depth, detailed Proposed Technical Approach (PTA) study that examined both centralized and decentralized systems. The PTA defined four viable alternatives. The Navy Ship Acquisition Manager (with PM2 approval) then selected a centralized computer system concept. (SE1)

Trident was the first Navy system to specify top-down design. The use of top-down design was addressed in the Command and Control System (CCS) Software Management Plan (Ref. 34). The requirement included the application of the procedure not only to the overall system but also independently to the major modules of the subsystem. (SE3)

The Trident Software Management Plan (Ref. 34) specified a disciplined set of programming practices, including:

1. Use of CMS-2Y language;
2. Use of structured programming by all developers;
3. Production of a software standards and conventions manual featuring design conventions, naming standards and conventions, coding techniques, etc.; and
4. Use of WS-8506 (Ref. 28) as a documentation standard. (IP2)

Trident planned for and implemented a Land-Based Evaluation Facility during the conceptual phase of development. It has the specific purpose of software development, test, integration, and certification. The facility includes tactical equipment as well as computer systems facilities. The facility supports total systems integration and verification of each tactical equipment suite prior to shipboard installation. (IP3)

#### 4.4.2 Pershing Weapon System

##### Description

The Pershing Ia is a mobile, nuclear ballistic missile system. The system includes all firing battery components required to conduct launch operations as well as equipment necessary for rear area support and maintenance functions. It is presently deployed in Europe.

Major equipment items required at a launch site are the erector launcher, power station, programmer test station (PTS), missile, azimuth laying equipment, and battery control central. The Pershing system contains two major computer systems. The PTS functions as the mobile fire control center. Each PTS can support three missiles, each of which contains a guidance and control computer (G&CC). Basic functions and computer characteristics for a Pershing Battery are shown in Fig. 4-13 and Table 4-17.

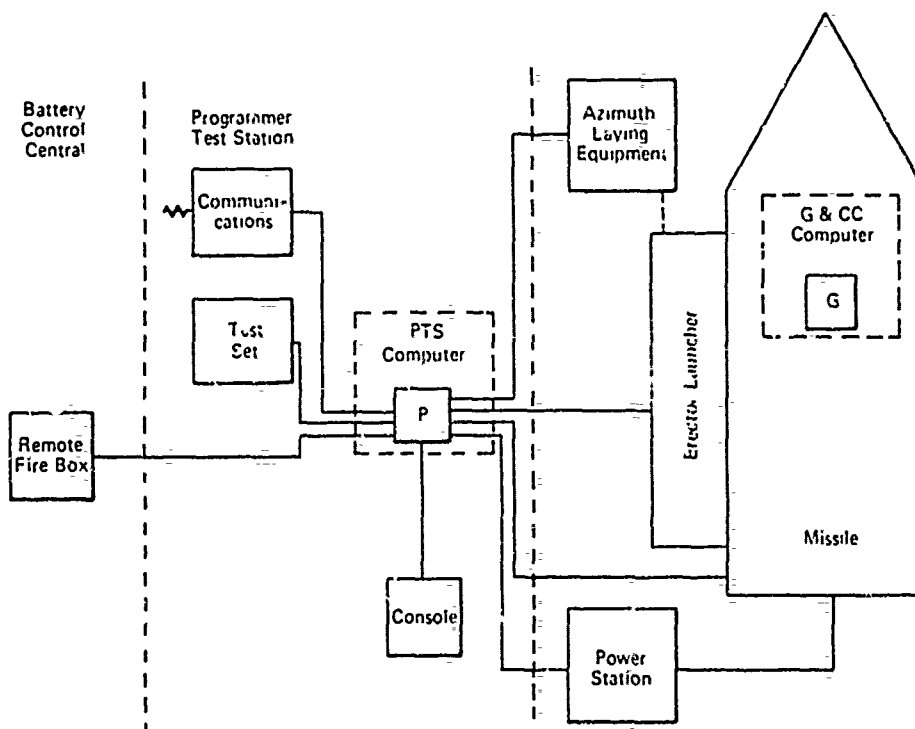


Fig. 4-13 Pershing Weapon System

TABLE 4-17  
COMPUTER SUMMARY

Unit	Type	Function	Processors	Memory
P	Burroughs D84 (24 bit, 4 $\mu$ s)	Target data entry, firing sequence control, missile preset, status monitoring	1	16k
G	Bendix BDX 820 (16 bit, 2 $\mu$ s)	Missile in-flight guidance and control	1	4k

### Acquisition History

The Pershing Weapon System was developed and deployed during the 1958-1964 period. The Pershing Project Manager's Office (PPMO) has controlled development and generated the System Specification. MIL-STD's-480 (Ref. 30) and -490 (Ref. 8) have been used to control configuration management procedures and documentation.

### Management Information

Program Status	Deployed
Program Manager	PPMO at MICOM, Redstone Arsenal, Ala.
System Contractor	Martin Marietta Aerospace (MMA)
Type Contract	Various over the years (CPFF, Engineering Support by Assigned Task), Maintenance (O&MA), R&D (CPFF) (CPIF), Production (fixed price)
Software Contractor	MMA - part of engineering task
Maintenance Agent	MMA - part of engineering support task.
Software Deliverables	Tape, specifications, documentation (flow charts, logic diagrams, etc.)
Validation Agent	PPMO Staff, Inertial Guidance Lab (IGL - Redstone), Change Control Board (CCB - PPMO, IGL and MMA Members). Program Manager has overall control.
Integration Agent	MMA with CCB and PPMO control.



### Highlights

An example of the PPMO technical control of computer hardware is the recent digital "Airborne Tactical Computer" development program. PPMO set the upper limit on the amount of core at 3200 words. When requirements forced the total over this number, tradeoffs were visible to the PPMO who then made the final decisions on what was to be cut back. (MP1, SE1)

The contractor does not use, in any important way, personnel designated as Programmers. Software is designed, developed, validated, and maintained by engineers who have the capability of programming. (IP2)

Two of Pershing software programs (airborne and countdown) are strongly controlled by PPMO who uses the DoD Configuration Management program established by DoD Directive 5010.19 (Ref. 35) and DoD Instruction 5010.21 (Ref. 36). Specifically, they set forth the requirement to use MIL-STD-490 (Ref. 8), Type B-5, specification to assure development of a computer program satisfactory for the intended use. After the baseline is established using MIL-STD-490, changes can only occur using guidelines in MIL-STD-480 (Ref. 30). This technique also generates a Mandatory Item Specification (MIS) as required by ASPR 1-202(a) (Ref. 37). PPMO elected to have only the two most critical of the over 149 software programs controlled by the above technique because of the costs involved in implementing the government standards. (AM1, MP3)

The PPMO has had a strong hand in Pershing hardware and software design, quality assurance, and maintenance. He has been able to keep this control by having an engineering staff available to him at MICOM (Redstone Arsenal). (MS1)

The PPMO has used computer software contractors as advisors on management techniques. (MS1)

The contractor indicated that, in the critical programs, the verification costs exceeded the programming costs by a 6:1 ratio. (TT1)

No attempt was made in Pershing to standardize computer hardware or languages used.

#### 4.4.3 SAM-D Weapon System

##### Description

SAM-D is an air defense Weapon System designed for use by the U.S. Army. It is an advanced surface-to-air guided missile system capable of multiple, simultaneous engagements in an ECM environment against high performance targets of the 1980-1990 time frame.

The system has two operational echelons -- Battalion Control and Fire Control Section. The Battalion level commander exercises operational control over and coordination of the Fire Control Section operations including rules of engagement, operational procedures, and management direction. The Fire Control Section performs all functions associated with the detection, track, identification, engagement, and destruction of targets.

SAM-D is currently in the Advanced Development Phase with emphasis on demonstration of guidance feasibility. Computers and computer programming structure are being developed to meet the specific needs of the SAM-D system. A description of the SAM-D system is shown in Fig. 4-14.

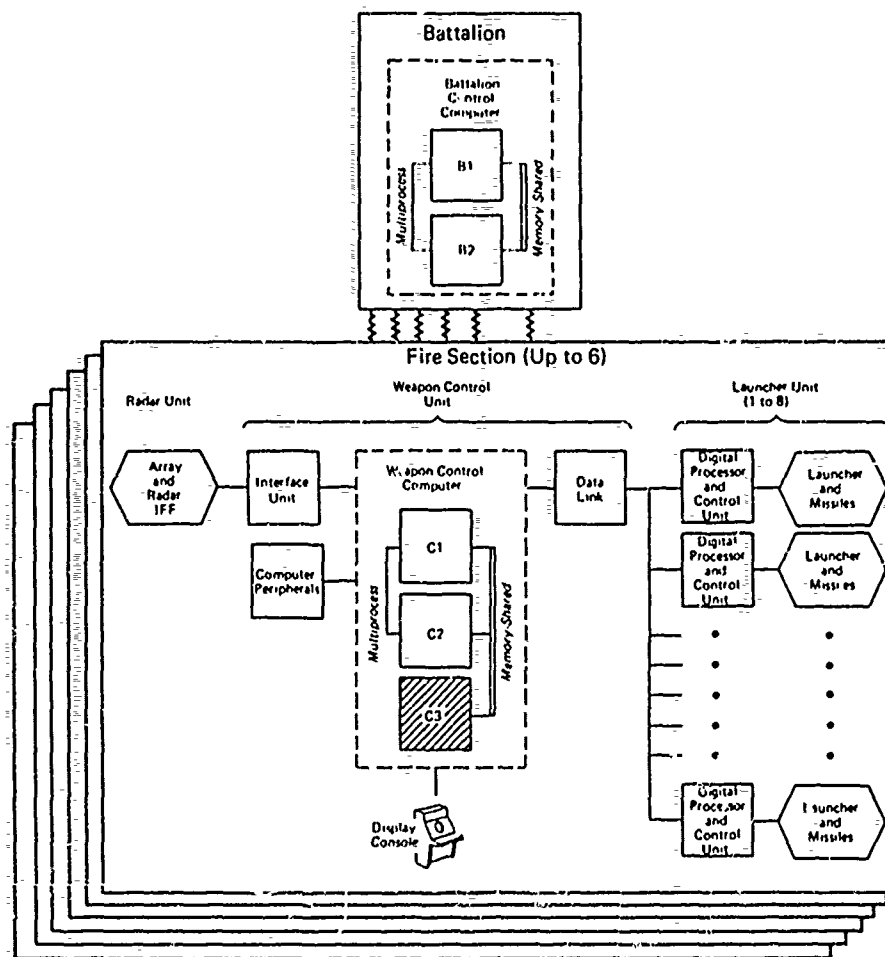


Fig. 4-14 SAM-D Weapon System

Weapon Systems functions and computer characteristics presently sized for SAM-D are shown in Table 4-18.

TABLE 4-18

COMPUTER SUMMARY

Unit	Type	Function	Processors	Memory
C1, C2	Raytheon (24 bit, 1 $\mu$ s)	Weapon Control Computer (multiprocessor): radar control, detection and track, communications, display, weapon assignment, status monitoring, missile guidance	2	160k (expandable to 256k)
C3	"	Future third processor	(1)	

Acquisition History

SAM-D is a descendant of the Field Army Ballistic Missile Defense System (FABMDS). FABMDS was intended primarily for defense against ballistic missiles. The FABMDS program was terminated in October 1962 when the Army determined that it was too complex and too costly.

Almost concurrent with this action, the Army Combat Development Command (CDC) proposed the development of an air defense system to replace Hercules and Hawk. This system would meet the 1970's air threat and have an Anti-Tactical Ballistic Missile (ATBM) capability. Technical feasibility studies were conducted in 1963 and 1964. In October 1964, the program was reoriented to become a combined component verification program and a study of design tradeoffs.

The SAM-D project was established in August 1965. The Request for Proposal (RFP) for contract definition was prepared and issued to 13 qualified bidders in April 1966. Three of the responding contractors were selected for fixed price contracts to complete contract definition of their proposals.

In May 1967, the Raytheon Company was selected to proceed with Advanced Development for SAM-D. In November 1967, the initial letter contract was definitized as a cost plus award fee contract for a period of 28 months.

The SAM-D Defense System Acquisition Review Council (DSARC) reviewed SAM-D on 4 February 1972, and a contract for engineering development was executed with the Raytheon Company.

In early 1974 a DSARC II review was conducted. At this review it was decided that the SAM-D system would enter a transition phase to develop validation data particularly for the Track-Via-Missile principle. Currently, the validation tests are being conducted at White Sands and a DSARC IIA review is tentatively scheduled for late 1975. Concurrently, work on the essential features of the engineering development single fire section model of SAM-D is proceeding at a reduced level.

Management Information, SAM-D

Program Status	Recycled back to Advanced Development from Engineering Development
Program Manager	U.S. Army Material Command, Redstone Arsenal
System Contractor	Raytheon Missile Systems Division, Bedford, Mass.
Type Contract	Cost plus incentive fee
Software Contractor	Raytheon
Validation Agent	U.S. Army TECOM
Maintenance Agent	U.S. Army
Software Deliverables	User's manual, maintenance manual, and program tapes
Integration Agent	Raytheon

### Highlights

Prior to Full Scale Development, the software requirements for the major SAM-D functions were delineated in a set of over 150 documents called the Data Processing System Requirements (DPSR's) prepared by the contractor. The DPSR's are reviewed by the Program Manager Office. Core and timing estimates are also made but these have much uncertainty. (MP1)

The PERT system of program scheduling has been used. The SAM-D Software Management Plan prepared by the contractor specifies in detail the milestones for software development covering analysis, design, implementation, integration, and testing. (AP1, AP2)

Extensive system tradeoff analyses were conducted by the contractor during the Engineering Development Definition Phase of SAM-D. The analyses included a study to determine the system functions allocated to the dedicated hardware of the radar unit and to the general purpose computer in the weapon control unit. (SE1)

The development of the software for the SAM-D system was identified as a high risk item during the advanced development phase. Provisions were made for expansion of the memory and CPU capability in the event that timing and sizing estimates for the software were inadequate. A top-down approach is employed in software development documents. (SE2)

The software and unique computer are undergoing parallel development at the prime contractor's facility. JOVIAL higher order language is being used to develop the computer instructions. For software checkout purposes, the SAM-D computer is emulated on a Univac 1108. (SE3, IP1)

As part of the software support tools for the SAM-D system, the contractor developed the following: a JOVIAL compiler and related assembler, various utility modules, data base, and operating system. The Program Manager Office has had difficulty in monitoring the software development because of the use of these nonstandard support tools and the use of a modified form of JOVIAL as a higher level programming language. (IP1)

Software integration is accomplished in the SAM-D Tactical Software Development Facility. The contractor and Program Manager felt that this type of a software test-bed facility should be encouraged for application to other programs by appropriate funding. (IP1)

Prototype and tactical software adequacy is verified with a test-bed simulator at the prime contractor's facility, followed by operational integration assessment at White Sands. (IP3)

The SAM-D Project Office has a staff of eight software specialists to monitor progress and to approve the prime contractor's plans with respect to software development. The Project Office has been assisted by IBM Federal Systems Division, which has provided an independent assessment of certain areas of the software development. (MS1)

## 5. SUPPLEMENTARY DISCUSSIONS WITH SERVICE AND INDUSTRIAL ORGANIZATIONS

### 5.1 SERVICE ORGANIZATIONS

In addition to the review of previous studies and discussions with specific Weapon System managers and contractors, several of the service organizations that are significantly involved in software system development were contacted. Included in this section are brief summaries of the results of contacts with the Navy Fleet Combat Direction Systems Support Activities (FCDSSA), the Rome Air Development Center (RADC), the Army Center for Tactical Computer Sciences (CENTACS), and the DoD Joint Logistics Commanders' (JLC) Software Reliability Work Group (SRWG). These groups cover a spectrum of activities from software development and operational support, through R&D in software technology, to a study of how software affects system reliability.

### 5.1.1 Fleet Combat Direction Systems Support Activities

As directed by the Chief of Naval Operations in OPNAVINST 3500.27B (Ref. 38), the "primary emphasis of Fleet Combat Direction Systems Support Activities is focused on producing and supporting high quality software for the Naval Tactical Data Systems (NTDS) family of systems (as installed in naval ships and aircraft, related systems) and such other Command, Control, and Communications Systems as may be directed." The Fleet Combat Direction Systems Support Activities (FCDSSA) are two separate commands colocated at Fleet Combat Direction Systems Training Centers, one at Dam Neck, Virginia, and one at San Diego, California (formerly known as Fleet Computer Programming Centers, Atlantic and Pacific). The mission of both activities as stated in OPNAVNOTE 5450 (Ref. 39) is "to plan, design, construct, test, and deliver Combat Direction Systems (CDS) computer programs for the operating forces, including training programs, as assigned; to correct, update, modify, enhance, and distribute operational programs in accordance with evolving Fleet requirements; to provide ancillary computer programs in support of such computer program development and maintenance; and to provide technical assistance and computer programs to the Shore Establishment, as directed." However, individual Weapon System project managers are responsible for interfacing to NTDS.

The workload of the two activities is generally divided by type of ship and aircraft. For example, FCDSSA(SD) is responsible for the CV and E-2C, while FCDSSA(DN) is responsible for DLG-28, DDG-9, and DLGN-38.

The FCDSSA's produce computer software using civilian computer programmers under a level-of-effort contract. Naval personnel are also used in the software design and implementation process, in an effort to ensure that the resulting programs are realistic for use in an operational environment. FCDSSA(SD) is the controlling agency for Navy compilers and has responsibility for the maintenance, delivery, support, and configuration control of these programs, as directed by the Chief of Naval Operations.

There is a continuing effort to expand the capability and improve the efficiency of programming techniques used by the Navy. New techniques and developments in industry and within the services are monitored for possible application to the support of Navy Combat Direction Systems (CDS).

FCDSSA(DN) has developed the Handbook for Program Development and Production Procedures of Digital Processor Program (Ref. 11), which describes 49 tasks to be accomplished during software development. The tasks are specifically related to the documentation to be produced in accordance with SECNAVINST 3560.1 (Ref. 33).

A conference at FCDSSA(DN) led to the following observations pertinent to this study:

1. Several systems are not properly documented when received for maintenance, and systems delivered with maintenance procedures were documented differently.
2. The most serious problems are generated at the beginning of development (e.g., requirements definition, system design, etc.).

In addition, there is a definite need for an integration agent responsible to the Program Manager with specified authority during development.

3. It was suggested that the separation of maintenance and development might remedy the continuous development syndrome of software.
4. International agreements, such as interoperability with NATO, do affect the technical content of software.
5. It was suggested that the "meta compiler" approach is a good way to support software implementation for the proliferation of computers.
6. FCDSSA's experience indicated that portability of software is needed but much work remains to make it practicable.

FCDSSA(SD) indicated views as follows:

1. The user (in this case, the Fleet Commander in Chief) must control the operational support as well as design, development, and testing of Combat Direction System Software.
2. The Combat Direction System of a ship should be separate from other functions. Sensor and Weapon functions are better handled by dedicated hardware and software.
3. The user should participate in the design and development of software. An advisory role is not sufficient.
4. The CMS-2 high order language is not generally used for executive, input/output, and special purpose high speed routines where it is inefficient.
5. Documentation should be considered a deliverable item and, hence, a part of the system acquisition contracts. In the past, this has not always been the case.

Testing activity at San Diego is divided into three major groups: Operational Test and Delivery, Simulation Test Support, and Evaluation and Analysis. The preferred approach to all three involves two phases, Quality Control and Quality Assurance. The Quality Control phase parallels contractor development. The primary activities are review of design documents, periodic reviews of program listing, issuance of software discrepancy reports to the project office, and receipt of preliminary library tapes for early testing by FCDSSA. Quality Assurance, which starts when the production library tape is delivered, is completely divorced from any design or development group and tests the delivered system against the defined requirements.

FCDSSA(SD) has developed a handbook for Technical Management and Life Cycle Budgeting of Digital Computer Data System Software (Ref. 40) that advocates the consideration of three classes of software. These are defined as

"1. Process Control: Class III. This is the earliest application of computers to industrial and commercial use. Automated tooling machines, fabric looms, and more recently radar transceivers and missile flight directors are examples of this class. There is no in-line operator interaction - the control functions being performed are entirely in support of, or as an integrated part of, equipment systems that run without necessity of human intervention during operations.

"2. Information Control: Class II. The possibility of permitting operator decisions to affect the sequence of computer processing itself on-line



resulted in a new kind of software application. Automated supply, accounts, and logistic systems were marketed for use by nearly every form of business, both commercial and military (ADP and MIS). Expensive machines were shared among many users or subscribers, each with the same or similar inventory-type inputs, and each with delivery or stock-level reports as outputs. Selection on-line is at the request of, and in response to, operator control actions, while the system operates on pre-determined record-processing programs.

"3. Tactical Control: Class I. With the advent of display consoles that provide dynamic presentation of tactical units and associated qualitative identity and operational status information with data feed-back, on-line, even the intimate man-machine relationships in systems of the FAA transient flight control type, and in combat information centers in ships and near battle areas, could be automated. However, in these applications the operator's interaction was essential for any meaningful system operation, and the full gamut of application difference can be shown in this class: with no equipment to serve, human operators are the source for all significant program requirements of input data itself, output data required, and processing algorithm selection, during system operation."

### 5.1.2 Rome Air Development Center

The Air Force Systems Command sponsors an extensive research and development program at Rome Air Development Center (RADC) in software technology. This center supports the development of techniques for all applications of computer systems, including Weapon Systems. An advance facility has been constructed to support in-house development as well as work by contractors from industry and other Air Force commands. Many existing tools have been installed at this facility for evaluation and are available to users.

The Information Processing Branch at RADC has active programs, both contracted and in-house, in four major areas:

1. Software error analysis and Quality Control,
2. Management control of software acquisition,
3. High order language control, and
4. Disciplined programming environments.

A Software Reliability Analysis Center (SRAC) is being established at RADC to collect error data from programs under development. Several reliability models are already available and will be tested. When the SRAC is established, it may be transferred to a software reliability organization analogous to the current hardware reliability organizations.

Several contracts are being let to collect management-related data. These data will be used to construct management tools. For example, a software cost study is being conducted, and the data from this study will be used to develop cost prediction models. Several contracts with software system contractors such as TRW, SDC, Hughes, and IBM are being let to obtain their production data from previous software development projects. An existing Programming Support Library as used by IBM is being purchased for study and use at RADC.

High order language control is proposed by providing compiler generation and validation tools. The tools provide a means to formally describe and validate languages and compilers. For example, the description of the language is checked for inconsistencies with a tool called Semanol, developed at TRW. Once the syntax has been established, a tool like JOVIAL Compiler Implementation Techniques (JOCIT) can be applied to construct the compiler, and a tool like JOVIAL Compiler Validation System (JCVS) can be used to test the compiler. Additional work is being done to collect and analyze data concerning the usage of language features and sources of error. It is expected that this effort will lead to better languages and compilers.

The disciplines being implemented at RADC are being made available to operational programs under development. Plans are being developed to attach the RADC facility to the National Software Works (NSW), which will make the RADC tools available to many users in industry as well as the Air Force. The structured programming guidelines developed for RADC by IBM are to be used by several Air Force programs such as PAVE PAWS and MIPS.

### 5.1.3 Center for Tactical Computer Sciences

The Army Materiel Command (AMC) has established a Center for Tactical Computer Sciences (CENTACS) with responsibility for research, exploratory development, and, on certain occasions, limited advanced development in the areas of tactical computer systems. The staff currently numbers about 375 professionals.

AMC guidelines for CENTACS stress the following: the conduct of R&D in tactical computer software and computer hardware, the development of software guidance documents and standards, the provision of consulting services throughout AMC, the operation and maintenance of a Teleprocessing Design Center, the conduct of advanced development for multi-application computers, the support of software and input/output devices, and the identification of unwarranted proliferation of computer hardware and software.

In APL discussions with CENTACS personnel, the following factors contributing to the problems of software development were identified:

1. Loose performance specifications,
2. Traditional software development phasing,
3. Software complexity,
4. Lack of visibility of the software product,
5. Poor documentation/unstructured software,
6. Low level language programming,
7. Marginal capacity hardware, and
8. Concurrent development of computer hardware and software.

CENTACS has made several suggestions which they believe will alleviate some of the problems. They recommend that throughout the total software development cycle, emphasis be placed on using fewer, more highly qualified software professionals. Sufficient time must be allocated initially for system formulation, analysis, and preliminary design with adequate consideration given to hardware/software tradeoffs. Provision must be made, concurrent with software design, for documentation, for the design of adequate tests, and for the planning of maintenance and enhancement capabilities. Similarly, a plan must be devised for an adequate Software Support System. Oversophistication must be avoided in both the design and implementation stages. As work progresses, effort should be made to incorporate already existing software (i.e., "don't reinvent software"). Flexibility and possible application to other systems can be enhanced by not prematurely introducing specific computer characteristics into the development cycle (i.e., maintaining machine independence). During the implementation phase, the use of high level languages and structured programming should prove helpful. As problems arise during the total development, the temptation to solve them with "quick fixes" (e.g., software patches) must be resisted. Performance measurement techniques should be employed at the appropriate stages. Throughout, all development decisions must be based on the total life-cycle cost of the software.

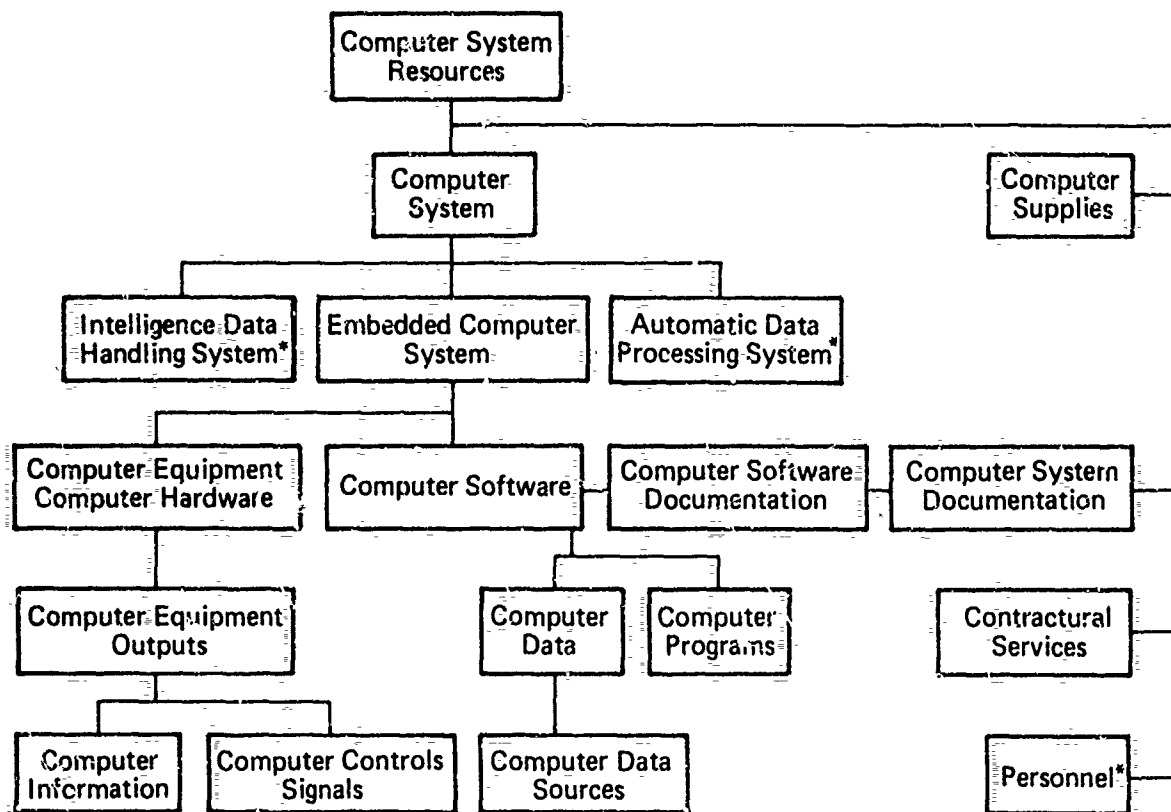
CENTACS is investigating some of these problems in three major areas: languages/compilers, real-time operating systems, and software engineering.

#### 5.1.4 Joint Logistics Commanders' Software Reliability Work Group (SRWG)

As part of a workshop on Electronic System Reliability, held at Arlie House, Virginia, 4-7 May 1975, a work group on software reliability has for over nine months been seeking methods for improving operational reliability of software. The Software Reliability Work Group (SRWG) reviewed acquisition and operational management techniques, documentation and testing, analysis, and design procedures that adversely affect the design, development, and production of electronic system hardware and software. The SRWG collected and analyzed 246 technical papers and reports and subsequently generated 43 specific program proposals for improved policies and procedures totaling almost \$40 million over the next eight years.

As one of seven work groups in the overall workshop, the SRWG consisted of fifteen permanent members and fifteen additional part-time technical specialists, with membership drawn from all three services, industry, and universities. In addition, each member solicited ideas from other organizations in both his functional and geographical areas.

APL has coordinated its Department of Defense Software Management Study with the JLC Software Reliability Work Group through its chairman, Lt. Col. John Manley (AFSC). The terms and definitions used in the recommendation sections of this report are those originally proposed by the JLC group to the DoD Steering Committee (see Fig. 5-1 and following definitions). The common usage has been adopted because the lack of commonly acceptable terminology in the past has caused considerable communication problems. With this set of definitions, the recommendations of the two groups can be directly compared. Preliminary comparison of draft reports indicates a good correlation between the JLC and APL recommendations. Twelve of the seventeen APL recommendations correspond to those made by the SRWG.



\* Not Defined

Fig. 5-1 JLC Digital System Definitions

DEFINITIONS PROPOSED TO  
THE DOD WEAPON SYSTEM SOFTWARE MANAGEMENT STEERING COMMITTEE  
FOR ADOPTION AS WORKING STANDARD DEFINITIONS IN  
THE DEPARTMENT OF DEFENSE

1. Computer: Electronic machinery which, by means of stored instructions and data, performs rapid, often complex calculations or compiles, correlates and selects data. Examples: Analog and digital processors, data processors, information processors, real-time control processors, electronic calculators, hybrid computers and communications processing.
2. Computer Equipment/Computer Hardware: Devices capable of accepting and storing computer data, executing a systematic sequence of operations on computer data or producing computer outputs. Such devices can perform substantial interpretation, computation, communication, control, or other logical functions. Examples: central processing units, terminals, printers, analog/digital converters, tape drives, disks, and drums.
3. Computer Supplies: Consumables designated specifically for use in normal operation of computer systems such as magnetic or paper tape, special forms, punch cards, printer paper, and ribbons.
4. Computer Software: A combination of associated computer programs and computer data required to command the computer equipment to perform computational or control functions.
5. Computer Program: A series of instructions or statements in a form acceptable to computer equipment, designed to cause the computer equipment to execute an operation or operations. Computer programs include operating systems, assemblers, compilers, interpreters, data management systems, utility programs, sort-merge programs, and maintenance/diagnostic programs, as well as applications programs such as payroll, inventory control, operational flight, satellite navigation, automatic test, crew simulator, and engineering analysis programs. Computer programs may be either machine-dependent or machine-independent, and may be general-purpose in nature or be designed to satisfy the requirements of a specialized process or a particular user.
6. Computer Data: A representation of facts, concepts, or instructions in a structured form suitable for acceptance, interpretation or processing by communication between computer equipment. Such data can be external to (in computer-readable form) or resident within the computer equipment and can be in the form of analog or digital signals.
7. Computer Data Sources: Devices, media, and associated actions that generate computer data for use by a computer system. Includes devices producing analog or digital signals; punched cards or magnetic tape; and associated procedures, processes, or methods used to initiate, modify, or terminate the operation of a computer system.
8. Computer Equipment Outputs: Computer data, computer control signals or computer information transmitted to any device or medium internal or external to the computer system.
9. Computer Information: The meaning assigned to computer equipment outputs by humans through the means of known conventions used in data representation.

10. Computer Control Signals: Computer equipment outputs in the form of electrical, optical or audio signals used to initiate, modify, or terminate the operation of non-computer devices external to the computer system.
11. Computer System: An interacting assembly consisting of computer equipment, computer programs, and computer data.
12. Embedded Computer System (ECS): An embedded computer system is a computer system that is integral to an electromechanical system such as a combat weapons system, tactical system, aircraft, ship, missile, spacecraft, certain command and control systems, civilian systems such as a rapid transit system, and the like. Embedded computer systems are considered different than automatic data processing systems (ADPE) primarily in the context of how they are developed, acquired and operated in a using system. The key attributes of an embedded computer system are:
  - a. It is a computer system that is physically incorporated into a larger system whose primary function is not data processing.
  - b. It is integral to such a larger system from a design, procurement or operations viewpoint.
  - c. Its outputs generally include information, computer control signals and computer data.
13. Computer Software Documentation: Technical data, including computer listings and printouts, in human-readable form which: (1) documents the design or details of computer software, (2) explains the capabilities of the computer software, or (3) provides operating instructions for using the computer software to obtain desired results from computer equipment.
14. Computer System Documentation: Information that describes the technical details of the computer system over its life cycle. Documentation includes, but is not limited to, equipment design specifications, engineering drawings, operators manuals, technical orders, computer software documentation, systems specifications, run diagrams, and interface specifications.
15. Computer System Resources: The totality of computer equipment, computer programs, computer data, associated computer documentation, contractual services, personnel and computer supplies.

## 5.2 INDUSTRIAL CONTRACTORS

To obtain a realistic understanding of software problems in the operating forces, APL arranged meetings with the users of selected Weapon Systems. To gain insight regarding the techniques and problems of the Weapon System designers, APL also held reviews with major Weapon System contractors. In addition, extensive visits and discussions were arranged with software and systems contractors. The overall objective of these interactions was a search for improved techniques, standards, and methodology to achieve cost-effective, error-free software.

The spectrum of reviews with industry covered the major phases of software management, design, implementation, verification, maintenance, and documentation. Recommendations were elicited from the industrial sector concerning the direction of present and future software R&D.

Table 5-1 is a matrix relating the APL recommendations to work done or recommendations by software and systems contractors. The boxes marked "X" denote contractor recommendations or support in the area indicated. It is interesting to note the emphasis on requirements, software visibility, and software tools.



TABLE 5-1  
APL RECOMMENDATIONS VERSUS SUGGESTIONS BY SYSTEMS/SOFTWARE CONTRACTORS

Recommendation	TRW	GRC	GRD	SDC	USC-ISI (NSW)	Boeing
<u>MANAGEMENT POLICY</u>						
MP1 Analysis and Validation of Systems Requirements	X	X	X	X		X
MP2 Software Visibility in Weapon System Acquisition	X	X	X	X		X
MP3 Software as Contract Deliverable	X			X		X
<u>ACQUISITION PLANNING</u>						
AP1 Milestoned Development Plan	X			X		X
AP2 Computer System Resource Development Plan				X		X
<u>SYSTEMS ENGINEERING</u>						
SE1 Systems Engineering of Computer Systems	X	X				
SE2 Provisions for Growth in System Requirements	X	X	X	X		X
SE3 Systems Engineering of Computer Software	X					X
<u>IMPLEMENTATION PROCEDURES</u>						
IP1 Software Development Support Tools and Facilities	X	X		X	X	X
IP2 Disciplined Programming	X	X				X
IP3 System Integration and Test Capability				X		X
<u>PROGRAM MANAGEMENT SUPPORT</u>						
MS1 Technical Staffing of Program Manager Organization	X			X		
MS2 Systems Engineering Agent	X			X		
MS3 Software Operational Support Agent	X			X		
<u>ACQUISITION MANAGEMENT</u>						
AM1 Standard Criteria for Weapon System Computer Resources Acquisition Management				X		X
AM2 Software Acquisition Guides						
<u>DEVELOPMENT OF SOFTWARE TOOLS</u>						
TT1 Software Test Tools	X	X		X	X	X

### 5.2.1 TRW Systems Group

#### Background

TRW Systems, Incorporated, with headquarters in Redondo Beach, California, is one of the largest software/systems contractors, employing approximately 12,000 people. They are heavily involved in design and software development activities for Army, Navy, Air Force, and commercial systems. TRW is the systems engineering support contractor for the Navy's ASW office and integration contractor for the Undersea Surveillance Program. TRW was the prime contractor on the Defense Satellite Program for the Air Force and for the Defense Satellite Communications System (DSCS II). Other major programs involving systems engineering and software activity include Minuteman, work for NASA on the Apollo-Soyuz and other space missions, the National Flight Data Center, and management information systems for the Energy Research and Development Administration. Typical of its large software activity is the \$100 million software development effort on the Army's Site Defense Program. This project has employed 300 programmers to produce 500,000 instructions of a 1,000,000-instruction computer program. TRW's disciplined software development methods have resulted in a record of on-time, in-budget Weapon System software development.

#### Activities, Tools, and Techniques

To support its Weapon System software development work, TRW has developed an array of software tools, procedures, design methodology, management controls, and verification methods. The software aids address a broad spectrum covering requirements, design, implementation, test and integration, documentation, and software management. The following subsections discuss these areas, including views and recommendations of TRW as well as key TRW activities in each category.

1. Software Requirements. TRW believes that many technical, cost, and schedule problems relating to software can be traced to inadequately defined requirements. They emphasize the importance of establishing traceability between requirements, specifications, and design. To this end, special requirements languages and automated tools to translate requirements into detailed specifications are being developed. Most DoD R&D money is applied to code and debug, which represent only 20% of the software problem. TRW feels more money should be spent on requirements, specifications, design, etc., which amount to 40% of the problem. The early products are major cost contributors to the 40% effort spent on test and integration.

2. Design. The approach to design is to complete both preliminary and detailed design before being committed to the implementation stage (code and debug). TRW says, "two-thirds of the time and one-third of the budget should be spent prior to coding." While top-down design was favored, the pitfalls were pointed out, such as the danger of deferring investigation of high-risk, low-level modules and deferring design and development of utilities. Typical software tools used in this phase are DACC, a design validation

aid; SPRINT, a quick-response simulation aid for computer system performance analysis; and FORMAN, a system for common data base management.

3. Implementation (Code and Debug). TRW's approach to implementation is that of continuous validation, i.e., "do a little, test a little." Highly disciplined programming procedures are stressed (down to mandating the maximum number of statements (100) in a subroutine). Numerous software tools are used in this phase, e.g., CODE AUDITOR, which checks for compliance with programming standards; UNIC, which verifies consistency of units for each parameter; and SINGAN, which automatically identifies singularities such as division by zero, square root of a negative number, etc.

4. Test and Integration. TRW's approach to test and integration is one of extensive testing, tight control, and clear accountability. A Unit Development Folder (available for customer review at all times) is maintained for each subroutine. It includes the test plan, test cases, and test results, as well as the schedule and final code listing. Integration is performed by an independent team. It involves checking subroutine interfaces, initially using dummy routines, which send headers but not data and which are gradually replaced with actual routines. Software aids used in this area cover routine/module/system testing, e.g., PACE, which monitors the specific part of the code that has been tested; EPIC, which enables modules to be linked together; and PPE, which identifies program areas that result in inefficient run times.

5. Documentation. TRW considers documentation an intrinsic and vital part of software development, not only to provide a necessary road map to designers, but also to provide adequate visibility for effective management and for future maintainability and expansion. The rationale is that of "anticipatory documentation," i.e., planning for and generation of documentation ahead of the phase that will require it. Typical software tools are FLOWGEN and AUTOFLOW, which produce flow charts from Fortran source code, and DOCGEN, which selectively extracts textual documentation from source code tape.

6. Software Management. TRW's software management philosophy is to use "sequential life-cycle planning," i.e., top-down planning, including risk analysis, planning for change, and completing each step so far as possible before proceeding to the next. A rigorous baseline configuration management procedure (Ref. 41) is stressed to maintain disciplined product control. Typical software tools used are GIM, a management information system with data storage, retrieval, and reporting capability; CHECKSUM, which identifies every element of the current software configuration; TCOST, which displays correlated budgeted and actual costs for each work breakdown structure element; and SPREAD, a computerized approach to estimating software development costs. Software costs and error data are extensively analyzed to determine how to improve the software process.

#### TRW Recommendations

1. More effort is required to clearly define requirements.
2. R&D support is needed, especially for front-end activity (requirement definition, specification analysis, and design).

3. An attempt should be made to quantify requirements uncertainties.
4. Language proliferation should be constrained because it increases software development costs.
5. Support software should be stipulated as a deliverable item.
6. Flow charts must be provided as part of documentation for assembly language programs, but not necessarily for well-structured higher order language programs.
7. A competitive contract definition phase, ending after preliminary design, is desirable whenever feasible.
8. MIL-STD's (e.g., MIL-STD-881, Ref. 5) do not emphasize software on a par with hardware in the Work Breakdown Structure. A constructive critique is needed here.

### 5.2.2 System Development Corporation

#### Background

The System Development Corporation (SDC), headquartered in Santa Monica, California, is a major software contractor. Its sales are \$100 million per year, and it employs about 3600 people. Over 85% of SDC's business is military or other Government work. It has no hardware affiliations and therefore claims complete objectivity in the selection or operation of a computer complex best suited for any given application. Typical examples of SDC's involvement in Air Force, Army, and Navy software activities are:

1. An \$11 million contract for its role as Computer Program Integration Contractor for the USAF Satellite Control Facility;
2. The Cheyenne Mountain Space Computation Center and the Tactical Information Processing and Integration (TIPI) System, also for the Air Force;
3. Research, analysis, development, and testing of the Parallel Element Processing Ensemble (PEPE) System for the Army Ballistic Missile Defense Agency;
4. Software support for the Navy's Ocean Surveillance Programs and a \$1.4 million contract for software maintenance support to the Fleet Combat Direction Systems Support Activity; and
5. Command Control and Tactical Systems Interoperability which includes support to: the Army Tactical Air Defense Systems (ARTADS) Program Office, the Ground and Amphibious Military Operations (GAMO) Program, and the joint service Tactical Air Control System/Tactical Air Defense System (TACS/TADS) Program.

#### Activities, Tools, and Techniques

SDC involvement in a wide range of software development activities over the years has led it to experiment with and apply many of the design methodology and software management techniques advocated in industry. While convinced of the value of software transferability, it still uses assembly language for some application programs. SDC sees standardization possibilities at the algorithm level, possibly in developing and using standard higher level languages for specific DoD application areas (e.g., C&C and MIS). It feels that another possible way to standardize is via firmware, i.e., through LSI technology. Thus, for example, one could develop standard LSI chips for navigation, guidance, or a wide variety of other applications. The company frequently uses the lead programmer concept as a forcing function but does not believe in disciplining programmers too rigidly.

Internal control procedures call for generation of a Software Program Development Plan as well as a Configuration Management Plan for planning and controlling projects.

While not touted as a panacea for all software problems, SDC is in the process of developing an interesting and potentially powerful software concept known as the "Software Factory". The objective of the Software Factory is to rigorously and systematically tackle the total software process (management, design, implementation, test, etc.) as part of an integrated

whole rather than as a conglomerate of ad hoc tools and techniques. The concept is indicative of SDC's views on the direction that software development and management should be taking.

The general goals of the Software Factory are to achieve a significant increase in software reliability, greater predictability, and a reduction in the implementation costs. The goals are to be achieved by providing a facility consisting of an integrated set of technical and managerial tools that will provide a disciplined and repeatable approach to software development.

The key problems to be addressed by the Software Factory are:

1. Lack of discipline and repeatability in the development process;
2. Lack of development visibility;
3. Changing performance requirements;
4. Lack of design and verification tools; and
5. Lack of software reusability.

While such techniques as structured programming, top-down program development, and production libraries are not claimed to be new, SDC believes it is their consistent use in an integrated manner, supported by an automated structure, that gives the Software Factory its greatest impact.

The Factory components consist of an integrated and extensible facility of software tools written in higher order language to enhance transferability to other machines. The requirement specification is linked step by step through the topcoat program modules to the successively lower level program modules. A key to implementation is the integration of the project management process into a top-down structure keyed to the design modules, with interrelationships, schedules, budgets, and other managerial information oriented toward this structure.

The data base consists of a software development data base and a project control data base. The user can access either data base via FACE (Factory Access and Control Executive), which provides access to program production library services. FACE also allows the designer to use a variety of development tools and gives management access to the project control data base through IMPACT (Integrated Management, Project Analysis, and Control Technique).

Typical of the technical tools provided are:

1. TOPS (Top-Down System Developer), a modeling tool that provides a method of describing and verifying a design as well as the data interface logic. It also permits replacement of modeled system components with real components as they are implemented;
2. TCG (Test Case Generator), an automatic technique for the design of test data. TCG determines the total network of statements in a program and helps ensure an adequate set of test cases;
3. PATH, a program flow analyzer that quantitatively assesses how thoroughly and rigorously a program has been tested;
4. AUTODOC, a tool to produce program and system documentation. It uses programmer comments and also reports on missing or incomplete comments; and
5. IMPACT, a means whereby the project is modeled and interfaced with the system model. IMPACT covers data base generation and

maintenance, project planning and control, and report generation. It is an automated tool that assists the manager by linking the requirements, activities, program modules, changes, and milestones into an integrated and dynamic management system.

#### Summary

The Software Factory is both a methodology and a set of tools that support the development and management of software systems. It is based on the concepts of disciplined programming, top-down program development, and program-production libraries, and it incorporates hierarchically structured program modules as the basic unit of production.

#### SDC Recommendations

1. Issue a number of contracts to industry for the development, in parallel, of complete and detailed standards for the total software process.
2. Develop capabilities to specify requirements concisely and unambiguously and assess their validity and completeness.
3. Develop formal terminology (including a glossary of terms) for specifying requirements in key DoD application areas.
4. The Government should promulgate an example of a standard, acceptable software Development Specification and a standard model of plans and procedures for testing software.
5. Require development and maintenance of software system architectural specifications early in the production cycle.
6. Every major software program should have a rigorous software program development plan, full multitiered specifications, and a configuration management plan.
7. Use "Contract Definition Phase" procurements whenever possible to improve cost reliability.
8. Begin configuration control of the allocated baseline immediately after the Preliminary Design Review (PDR).
9. Devise processors that support system building and simulation.
10. Standardize and disseminate algorithms by including them in the Request for Proposal (RFP) (e.g., navigation, tracking, guidance, etc.).
11. Use high level language for support software and, where practical, for application programs. Use assembly language only for time-critical or highly hardware-dependant programs.
12. Achieve standardization through firmware module standards, i.e., using LSI chip technology (e.g., a standard navigation chip).
13. Use a "red team" to review the "blue team's" output and documents.
14. Emphasize software transferability and reusability in R&D studies.
15. Implement a research and analysis program to provide a basis for personnel classification standards.

### 5.2.3 Boeing Aerospace Company

#### Background

The Boeing Aerospace Company of Seattle, Washington, has a software and computer systems engineering department of over 300 people. The group is primarily involved in the development of Weapon System software, although it has developed software for rapid transit systems and power utility applications. In addition to providing the software personnel staff for Weapon System programs, the team maintains a permanent group working on software IR&D and on the development of software tools and techniques. The department also acts in an advisory capacity to project managers, performing an independent audit function on programs. About two-thirds of Boeing's software effort is performed in-house. The company has developed computer software for several major programs, including the B-1 avionics systems, the Airborne Warning And Control System (AWACS), the Advanced Airborne National Command Post (AABNCP), the SRAM short range missile, and the Air-Launched Cruise Missile (ALCM). It has also done several software design analyses for the Canadian Long-Range Patrol Aircraft (LRPA). The Boeing Computer Services Company, which is independent of the aerospace company, also maintains a large staff involved primarily in scientific and business software and provides support to the Boeing Aerospace Company on a contract basis.

Boeing, as well as several other prime contractors, has developed a strong computer systems engineering software capability. The company was interviewed, as part of the Department of Defense Software Study (DSS), to obtain the viewpoint of this category of systems contractors.

#### Activities, Tools, and Techniques

Software activity at Boeing has ranged from small programs such as the Army's Radio Frequency Simulator System (10,000 instructions), to the large 290,000-instruction AWACS program. Programming languages have included JOVIAL, Fortran, AED, and assembly languages for a variety of commercial and military computers.

To support its software work, Boeing is doing R&D in areas such as real-time systems structured design, computer-aided software design, automated software verification, and reusable support software.

The software design philosophy stresses a top-down process and design completion before coding. While insisting on disciplined programming procedures, Boeing favors a broad interpretation of the term "structured," and is cautious on "structured programming" for real-time systems.

The group is responsible for the development and maintenance of a support programming system, which supports all operational software development with assemblers, link editors, diagnostics, etc. Their Functionally Oriented System Simulation (FOSS) is an important design tool. It permits a software system to be modeled on a high level, allows iteration of design parameters, and checks the resulting system performance. It also checks processor loading and identifies the degree of exercising of the various program branches and subroutines.

Boeing has done extensive work in the development of software standards, including software design standards, Quality Assurance, documentation,



identification and release, and management standards. Rigorous control of software modules is maintained during the integration and test processes. As soon as a software module is released, it comes under a formal change control procedure, which is analogous to the hardware engineering change procedure. This dovetails with the independent technical audit approach used for the validation of Weapon System software.

Use of an integration and test facility is standard operating procedure for all major Weapon System software development. Testing starts at the individual instruction level. Computer simulation is then used to exercise the operational program in a modeled "real-world" environment, and actual equipment is finally connected to exercise the software in a dynamic "hot-bench" environment.

#### Boeing Recommendations

1. Unified interservice and interproject software acquisition standards are required.
2. More specific definition of requirements is needed. The requirements should be the minimum necessary to meet approved objectives.
3. System requirements should be validated during preliminary design (possibly using a contract definition phase). Modeling should be stressed at both functional and detailed levels to validate design concepts. Modeling results should be a deliverable item.
4. A milestone software Quality Assurance plan should be specified.
5. A comprehensive milestone documentation package, including "design to" and "as built" documentation, should be specified.
6. An independent audit function is very desirable.
7. Early agreement on design details is needed (timely and comprehensive Design Reviews).
8. Software external interfaces (hardware to software) should be "frozen" after the software Preliminary Design Review (PDR).
9. An Integration and Test Facility is required for software verification.
10. Support software should be stipulated as a deliverable item, preferably in a rehostable higher order language (HOL) to permit transfer to another host machine.

#### 5.2.4 USC Information Sciences Institute

##### Background

The Information Sciences Institute (ISI) of the University of Southern California (USC) at Marina del Rey, California, is coordinating the first phase development activities of the National Software Works (NSW). This is a project sponsored jointly by the Defense Advanced Research Projects Agency (ARPA) and the Air Force. The goal of NSW is to improve the productivity of the DoD software system building process. This is to be achieved by making advanced tools developed and used by the research community available to production staffs on a nationwide basis. The concept will utilize the distribution capability of the existing ARPA network, making available a large and extensible library of software development tools that are contributed and leased by industry.

Planning for the NSW began in the summer of 1973, with development starting a year later in July 1974. Developers of the NSW include the Massachusetts Computer Associates, Applied Data Research, Bolt Beranek and Newman, Massachusetts Institute of Technology, SRI, University of California at Los Angeles, and the USC Information Sciences Institute. Massachusetts Computer Associates will assume the role of system integration contractor on 1 July 1975. As capabilities become available, they will be tested by the Air Force Data Systems Design Center (AFDSDC) at Gunter AFS, Alabama, and by the Air Force Data Services Center (AFDSC) in the Pentagon. During FY 76, ARPA intends to initiate an effort to demonstrate the applicability of NSW concepts to Weapon System software.

##### Description of the National Software Works

The NSW is to be a software system that integrates several dissimilar computers on the ARPANET (Fig. 5-2) into a distributed software factory. It will serve as a model software production facility for the large number of DoD projects where software must be developed and maintained by geographically dispersed agencies and contractors or where software development and maintenance tools will not run on the target hardware. The techniques for interfacing computers and tools will be applicable to confederations of dissimilar computers in a single location as well as to geographically distributed machines. The NSW will supplement conventional software libraries and will provide centralized management of the large inventory of available tools.

The following are problems that have held back the use of software tools and which NSW is designed to alleviate:

1. Users are frequently unaware of existing tools.
2. Tools written for one machine do not work on another.
3. Software development tools tend to require extra hardware resources.
4. Proprietary tools leased on a per installation basis result in high cost thresholds.
5. Many software development tools require interaction with the programmer, hence there is a need for an interactive system.



**Fig. 5-2 ARPA Network, Logical Map, January 1975**

NSW uses the ARPANET as a distribution system but automates the peculiarities of the ARPANET components, making them transparent to the user. The controlling software of the NSW is the Works Manager, which will provide

1. Automatic movement of files,
2. Access to tools and files,
3. Standardized user interface and protocol,
4. Coordinated accounting,
5. Policy enforcement, and
6. A directory of tools.

Users will access the NSW via interactive terminals. Tools will run on machines selected by the tool supplier (industry or Government agency). Input files requiring access to the tools will be moved automatically by the Works Manager over the network to the appropriate machine. While most other networks tend to interconnect relatively homogeneous hardware and compatible software, NSW will tie together, via the ARPANET, a wide variety of hardware and software using standard protocols. It will act as the "yellow pages," helping the user locate products and tools that meet his requirement.

Phase I development has the limited objectives of demonstrating the capabilities of the file editor and secretary, documentation production, and the remote job entry. Phase II will encourage industry to contribute software tools, such as Meta Cobol and the CMS-2 compiler, and possibly to emulate some processors, such as the AN/UYK-20.

In the future, control mechanisms will support sophisticated tools for the specification and implementation of management policies of project control. These policies typically will specify who may access and/or change modules, what cross checks must be carried out whenever a module is changed, and so on. Policies are to be explicitly separated from the implementing mechanisms so that two organizations with different management procedures can share the same system and tools.

The initial version of the NSW is being implemented on the ARPANET. The accessibility of the ARPANET to DoD organizations should increase significantly in late 1975 when the Defense Communications Agency begins operating it as an unclassified service.

There are major technical, organizational, and procedural issues yet to be resolved. However they appear solvable, and a successful NSW should be a powerful weapon in our software arsenal in the years ahead.

## 6.1 MANAGEMENT POLICY

### 6.1.1 ANALYSIS AND VALIDATION OF SYSTEM REQUIREMENTS

MP1

#### Recommendation

Direct that a comprehensive analysis and definition program be carried out on software (as well as hardware) elements of each new major Weapon System during the Program Validation Phase, prior to approval of Full Scale Development. The software definition should be carried down to the level of subprograms performing major functions.

Cost estimates for the development and integration of each subprogram should be based on analysis, simulation, modeling, or construction of its principal parts, as called for by their respective newness or criticality.

#### Primary Problem Areas Addressed

- Inadequate System Analysis
- Lack of Requirements for Adaptability
- Complex and Excessive Requirements
- Unrealistic Cost and Schedule Estimates

#### Discussion

The most crucial factor in determining whether or not a given system is ready for Full Scale Development is the validity of its requirements, in terms of completeness, clarity and realism. In the system acquisition cycle, the final formulation and validation of requirements should occur during the Program Validation (Advanced Development) Phase. For many systems the Advanced Development Phase has concentrated on the design and demonstration of hardware subsystems or components, it being assumed that demonstration of software designs is not necessary at this stage. Actually, Weapon System performance is critically dependent on digital subsystems and associated software, and if these elements are not given full attention from the outset of the system acquisition process, penalties are almost certain in terms of increased costs, delayed delivery, compromised performance, and needlessly burdensome life support problems. Many examples of this situation are observable today.

It is impossible to establish the realism and adequacy of requirements for a new complex system on the basis of paper analysis alone. The environmental conditions are never so simple that they can be expressed as a set of parameters, and the functions of operator performance cannot be represented by an equation. Instead, analysis must be supplemented by modeling, simulation and, in critical cases, by actual design and building of portions of the system. These methods must be applied selectively, and with judgment, to achieve an economic but adequate analysis and validation of system requirements.

Development of Software Requirements. Operational computer programs constitute one of a number of highly integrated components of a Weapon System; a combat suite on a major force element such as a ship or bomber aircraft may consist of several Weapon Systems. Thus, the requirements for a particular computer program must be developed as part of a hierarchy of higher level requirements.

#### 5.2.6 General Research Corporation

##### Background

General Research Corporation (GRC), located in Santa Barbara, California, is a multidisciplinary organization with approximately 1300 employees. Principal products and activities include concept formulation and test-bed studies, war games, real-time simulation systems, automated validation tools, and language development.

GRC has been heavily involved in Ballistic Missile Defense (BMD) system simulations. It developed the Bag Attack Game (BAG) series, which simulates both area (exoatmospheric) and terminal (endoatmospheric) engagements. The company has also done major simulation work in the fields of transportation, radio communication, and command and control. In the field of computer program validation, GRC has produced the automated Program Validation System (called RXVP) and is under contract to the Air Force to develop an automated verification system for programs written in JOVIAL.

##### Activities, Tools, and Techniques

GRC is probably best known for its work in the development of test-beds, which are computer simulations to interactively develop and test real-time software systems, and for its work in program validation.

GRC has modeled and simulated numerous aspects of the complex phenomenology associated with BMD engagements. Sixty-eight of these BMD engagement simulations are current, and many more were developed and used over the past 10 years. The Advanced Real-Time Software System (ARTISS) is typical of the real-time software simulation work done for BMD site defense. This simulation includes a real-time operating system and tactical routines including surveillance, acquisition, confirmation, and interceptor guidance. Innovative techniques were developed to build and test this complex software, including a special higher order language (HOL), ENLODE, which enhances Fortran in the areas of data management, real-time control, and process construction.

GRC has developed test-beds for the evaluation of candidate computer systems for given requirements, including Simulation for Analysis of Computer Systems (SACS) and System Environment and Threat Simulation (SETS). These techniques have potential bearing on the Weapon System computer selection and sizing process.

RXVP provides increased thoroughness in the testing of Fortran programs. A major objective is the automatic identification of test-case requirements to ensure full-coverage testing.

The company is doing significant work in the area of validation research, including

1. Testing methodologies: rigorous and detailed definition of test plans, systematic control of testing, and measurement of testing effectiveness;
2. Languages: error-resisting languages that require the programmer to supply additional information about actions intended for each program module; and

3. Structured programming: advanced techniques of software production to increase software system reliability and maintainability.

GRC Recommendations

1. Define guidelines for the procurement of data processing systems that encourage growth margins in the computer hardware capability of providing more flexibility in software design, such as permitting the use of HOL's.
2. Require specification of (and provide funding for) an acceptable level of software testing and exercising in software development procurements.
3. Plan for software breadboarding before software production (especially in novel areas).
4. Support software development techniques that make software inherently more testable (e.g., structured programming, error-resistant languages).
5. Support R&D for the development of automated testing techniques (including automatic generation of comprehensive test cases).

## 6. DISCUSSION OF RECOMMENDATIONS

The purpose of this Section is to discuss in more detail the recommended actions presented in summary form in Section 2 of this report. The briefs in Section 2 state each recommendation together with the problems it addresses, actions necessary to implement it, and brief remarks in support of the recommended action. In this section each recommendation is discussed to clarify its nature, elaborate the rationale behind it, and cite specific supporting findings drawn from the review of previous studies, Weapon System reviews, and service/industry visits. For ease of reference, each discussion is preceded by a restatement of the recommendation and followed by a statement of the proposed implementation.

Problem Areas Addressed by Recommendations. Section 1.3 describes a diagram (Fig. 1-1) that shows the cause/effect relationships of the most frequently cited problem areas in Weapon System Software Acquisition Management. This chart has been found to be useful in helping to understand the many problems encountered and how they relate to the principal phases of the system life cycle. Table 6-1 has been prepared to assess how the actions recommended in this report address the principal problems identified.

The columns in Table 6-1 correspond to individual problem areas taken from Fig. 1-1, with those from each column of the figure listed from top to bottom, and the groups separated by acquisition phase. (The three blocks under "Inputs" in Fig. 1-1 are not included in Table 6-1 because they are inherent aspects of software and are not subject to control.) The rows of the table correspond to the recommendations of this report. Recommendations expected to have a direct effect on a given problem area are denoted by a "D" in the appropriate column, and those believed to have an indirect effect by an "I."

In examining the pattern of entries in Table 6-1, it is evident that each recommendation typically affects two to four problem areas directly (D) and a greater number indirectly (I). This cascading effect is to be expected from the nature of the interrelation among problem areas, as depicted in Fig. 1-1. It is necessary to refer to the figure to understand the total effect of any one recommendation; interested persons may find it useful to derive their own conclusions relating to the indirect effects of each recommendation by reference to the "road-map" in Fig. 1-1.

Distribution of Problem Areas Addressed. The general distribution of D's and I's in Table 6-1 shows that almost all the D's occur prior to production and deployment, and the majority are found prior to the Full Scale Development phase. In part this is a reflection of the general conclusion that the most effective actions to control software development must occur during the formative stages. However, in considerable measure the high population of D's in the section under Program Validation is due to the fact that the problem areas are listed where they first present themselves. Most of these problem areas continue to have direct impact well into Full Scale Development.



An additional point that should be emphasized is that the indirect results of solving problems during the beginning of development are as important, if not more so, than the immediate results. Thus while only one of the specific recommended actions is shown to impact on costs directly, all are actually aimed at positive measures to minimize one or more aspects of system cost.

It may be noted that two columns in the table have no entries. The one labeled "Increasing Demands on System Software" is a direct result of the external inputs, "Complex and Growing Threat" and "Advancing Computer and Sensor Technology," which are not subject to control. However, the impact of that problem on actual software system requirements is directly addressed in recommendation MP1. The other empty column is entitled "Lack of Methods for Sizing Software Tasks." This area is susceptible to remedial action, but a specific approach has not been recommended at this time. It is included among subjects covered in Section 7.

In the textual material in this Section, a paragraph headed "Primary Problem Areas Addressed" lists those problems that are directly addressed by each recommendation.

Table 6-1  
Software Life-Cycle Problems Addressed by

RECOMMENDATION		CONCEPT FORMULATION						PROGRAM VALIDATION															
		INADEQUATE SYSTEMS ANALYSIS	INCREASING DEMANDS ON SYSTEM SOFTWARE	INSUFFICIENT SOFTWARE BY MANAGERS	LACK OF SOFTWARE UNDERSTANDING	LACK OF VISIBILITY FOR ADAPTABILITY	COMPLETIBILITY REQUIREMENTS	UNDEVELOPED EXCESSIVE CONTROL TECHNOLOGY	LACK OF SOFTWARE SYSTEMS ENGINEERING	LACK OF POLICY GUIDANCE AND PLANNING	INADEQUATE SOFTWARE PROCESSORS FOR PROGRESSOR CAPACITY	INADEQUATE ARCHITECTURE SOFTWARE TRADEOFFS	NON-IDENTICAL SOFTWARE DEFINITION	INADEQUATE ARCHITECTURE MANAGEMENT	ESTABLISHED PROCEDURES	UNSPECIFIED STANDARDS	UNSPECIFIED SUPPORT PROVISIONS	UNSPECIFIED MAINTENANCE SCHEDULE ESTIMATES	NO PROVISION FOR CHANGING REQUIREMENTS	PROGRESSOR CAPACITY	UNWARRANTED BOTTOM-UP DESIGN	PROGRESSOR CAPACITY	
MP1	Analysis and Validation of Systems Requirements	D				D	D				I		I					D	I	I		I	
MP2	Software Visibility in Weapon System Acquisition			I	D				D					D	I			D					I
MP3	Software as Contract Deliverable				D																		
AP1	Milestone Development Plan								D					D				D					D
AP2	Computer System Resource Development Plan													D	D	D					I	I	
SE1	System Engineering of Computer Systems								D			D	D							I	I		
SE2	Provisions for Growth in System Requirements					D					D								D				
SE3	Systems Engineering of Computer Software												D								D	D	
IP1	Software Development Support Tools and Facilities																						
IP2	Disciplined Programming														D								
IP3	System Integration and Test Capability																						
MS1	Technical Staffing of Program Manager Organization								D	D						I	I	I		I			
MS2	Systems Engineering Agent								D			D	D					D		I			I
MS3	Software Operational Support Agent														D	D							
AM1	Standard Criteria for Weapon System Computer Resources Acquisition Management								I	D		I	I	I	I	D	D		I				
AM2	Software Acquisition Guides			D					I	D		I	I	I	I			I			I	I	
TT1	Software Test Tools																						

D - Direct Correlation  
I - Indirect Correlation

### Problems Addressed by AFL Recommendations

13

The application of this process to a major new combat system can be illustrated with reference to Fig. 6-1. This figure represents the development of requirements for a complex, multifunction combat system that is representative of the shipboard systems reviewed in this study. As a minimum, the Combat System Requirements will state the ship's mission, operational requirements, major configuration constraints, and operational support concepts.

The Combat System Requirement Analysis involves the development of a combat system configuration in terms of separate but interrelated systems. Each separate system may embody computer processing or control. In any case, the overall requirements must be broken down into those for individual component systems at this stage, with clearly defined relationships and interfaces.

The System Requirement Analysis for each constituent system is concerned with such decisions as tradeoffs between special purpose digital hardware and programmable general purpose computers, selection of computers by type, memory, size, peripherals, and special approaches such as multiprocessing. This second level of computer requirements analysis should be done in conjunction with the architectural design of the special purpose hardware, general purpose computers, and computer programs. Insofar as the computer software is concerned, this phase results in a set of requirements for each program and a definition of its external interfaces with the other portions of the system.

The third stage, Software Design Analysis, is the breakdown of each computer program into the primary functional components or modules, and sufficient modeling, simulation, or actual preliminary design of critical functions to produce reasonably well-founded estimates of time and core allocations. This step results in a statement of Design Criteria for each computer program.

System Design and Iteration of Requirements. While the above process is referred to as "analysis," it is seen to go step-in-step with the process of configurational design. At each stage the definition of the system must become one level more detailed and each functional block subdivided into a number of separate but integrated components. Only in this way can the overall requirements be translated into a set of requirements on individual subsystems with assurance of completeness and consistency. Further, at each stage there must be feedback of any problem areas to previous stages, to permit relaxation of unnecessary constraints or excessive performance goals in the interest of reliability and cost.

All of the above steps in the definition and validation of the system requirements should take place before the formal DoD DSARC II review. They must be completed in order to provide a basis for life-cycle costs, identification of development risks, and determination of the realism of the plan for Full Scale Development, as required in DoD Directive 5000 (Ref. 4).

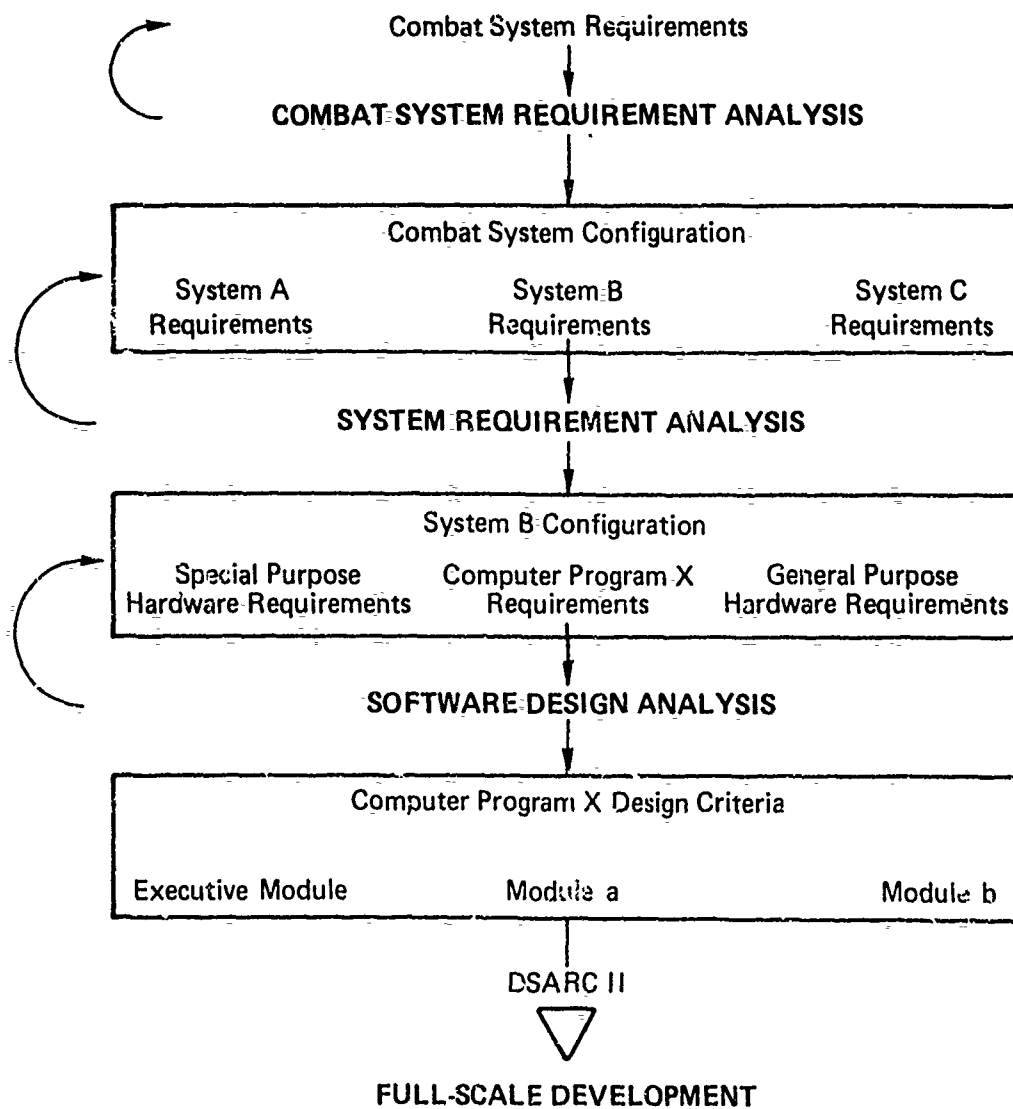


Fig. 6-1 Development of Requirements and Design Criteria

Application to Evolutionary Systems. The application of the above discipline must, of course, be tailored to each specific system, its degree of novelty and complexity, and development history. For many systems, both ship and airborne, developments have borrowed from the specifications and design of existing programs, with modifications as required to meet new missions, capabilities, or threats. This evolutionary approach carries benefits of minimizing new developments. On the other hand, it carries dangers in that the actual requirements for the new unit may not be fully thought out and developed. A number of the programs which reported "growth," and hence overrun of computer resources during or after system development have been of the evolutionary type.

Experience in Industry. There is broad agreement among organizations engaged in development of system software that inadequately defined requirements constitute the greatest single source of difficulty in software development. An excellent discussion of this problem is contained in a paper entitled "Software Requirements Analysis, Sizing and Costing" presented at the TRW Software Workshop (Ref. 42). The following are three pertinent excerpts from this paper:

1. "Often the difference between success and failure of a large software project lies in the consistency and completeness with which the system requirements have been specified, and in the fidelity with which these requirements have been translated into estimates of the cost, schedule, and hardware required to support the application."
2. "If there is no statement of requirements the first step in software development will be design - there is no other alternative."
3. "The most difficult act in producing software is to eliminate the errors. The most important step in eliminating errors is a thoroughgoing test program. The only basis for a thoroughgoing test program is a set of requirements all of which must be testable. If the requirements are incomplete or vague, the test program will be incomplete; if the requirements are non-existent the test program is in a shambles."

Representatives of the Systems Development Corporation made the following related recommendations:

1. "Much more effort and money should be expended on the preparation of good development specifications early in acquisition. The Government should be an active participant in the technical effort leading to these specifications."
2. "PDR should be a specification concurrence. The procuring agency and the contractor who have jointly prepared the development of specifications should work as allies not as protagonists. Trainers, maintainers, and users should have made their input prior to PDR."

These suggestions were made on the basis of experience that, until preliminary design of the software has been developed, appropriate iteration

of system requirements costs and schedule estimates are unrealistic. This experience was also generally reflected in discussions at Boeing and at other contractors.

Experience in Weapon Systems. The overriding importance of adequate top-level requirements was strongly stated in discussions at FDCSSA, San Diego. The E-2C Program Manager also commented on the effect of inadequate requirements on system costs.

In the P-3C and S-3A aircraft systems development programs, a long-term analysis effort by NADC was the basis of contracted developments for specific aircraft. NADC not only developed requirements for these systems but also built laboratory models which demonstrated feasibility and confirmed the computer resource needs.

The AEGIS program is an example of a total system development starting with an in-depth assessment of requirements. This development involved a funded requirements study by seven contractors, a comprehensive Navy assessment study, commonality studies with concurrent Army development (SAM-D), advanced development projects which included a feasibility demonstration of multifunction radar computer control, competitive contract definition, and a two-phase engineering development program.

Timing. A most critical point of the recommended requirements analysis and validation effort is that it must be carried out early enough to enable reassessment of software requirements. This will expose the cost of obtaining a desired level of performance capability and allow requirements to be scaled down in areas where the expected performance does not justify the cost. Commitment to Full Scale Development can then be made within controllable bounds of cost, schedule, and performance expectation.

#### Implementation

Amplify and extend the definition of requirements for the System Design Review (SDR) to include review of the functional definition of each software program, its modular configuration, and its evaluation in terms of core and timing allocations in order to show that it will meet performance requirements within specified constraints (similar to Milestone 1 in SSD 61-47B, Ref. 3).

Amplify the requirements of the DSARC II review in DoD-5000 (Ref. 4) with respect to what is expected to be accomplished in the Program Validation Phase.

## 6.1.2 SOFTWARE VISIBILITY IN WEAPON SYSTEMS ACQUISITION

MP2

### Recommendation

Increase the visibility and understanding of major software components of Weapon Systems by putting them on a par with hardware components. This is to be done in terms of configuration control items, DSARC II reviews, design reviews, and other aspects of acquisition management.

### Primary Problem Areas Addressed

- Lack of Software Visibility
- Lack of Policy Guidance and Planning
- Inadequate Management Procedures
- Unrealistic Cost and Schedule Estimates

### Discussion

In order to increase the visibility of major software components in Weapon Systems, it is necessary to modify a number of basic policies, procedures, and standards that are used as approaches and references in contracting and managing systems acquisition. In particular it is important that policy in the Office of the Secretary of Defense (OSD) and in the services recognize software as a Configuration Item, to be delivered on a schedule, rather than consider it as an item of data. Software should be treated in a manner similar to hardware rather than as documentation. This will require modifications of policy documents addressing software as data items in DoD and in Armed Services Procurement Regulations.

Work Breakdown Structure. Many of the actions recommended in this section deal with updating of acquisition management procedures and standards, which will have the effect of increasing software visibility. However, a particular point of reference which needs suitable modification is the Military Standard on the Work Breakdown Structures for Defense Material Items, MIL-STD-881 (Ref. 5).

The specific work breakdown structures and definitions are contained in the Appendices to MIL-STD-881. These appendices define the levels of breakdown for each of several types of systems such as missiles, ordnance, space systems, and electronics. Computer programs do appear as an item, but only under electronics systems along with sensors, communications, data displays, and auxiliary equipment.

As a minimum, major computer programs need to be given a place comparable to major subsystems and not confined to a subhead under electronics systems. Perhaps the level should also be modified, but this question requires careful study.



Breakout of Software Costs. The work breakdown structure for software is particularly relevant to the breakout of software costs, which has long been a point of difficulty. A consistency among standards, practices, and contractual requirements is very important to avoid misunderstandings and problems in the course of development.

Implementation

One specific step which should be taken is to generalize and revise Appendix B (Electronics System) of the Military Standard - Work Breakdown Structures for Defense Material Items (MIL-STD-881, Ref. 5) to define a software subsystem category on a par with a hardware subsystem, along the lines being considered by the Joint Logistics Commanders' Software Reliability Work Group. Construct a suitable list of computer resources that should be covered during review processes. Other directives and standards for hardware should be studied by DoD to find similar transferability of their principles to software.

### 6.1.3 SOFTWARE AS CONTRACT DELIVERABLE

MP3

#### Recommendation

Specify that major computer software involved in Weapon System development be designated Configuration Items (CI's) and deliverables during Full Scale Development. This would generally include computer programs and computer data for

1. Operational software,
2. Development support software, and
3. Test and integration software.

#### Primary Problem Areas Addressed

Inadequate Interface Management  
Lack of Software Visibility  
Inadequate Documentation  
Lack of Transferability

#### Discussion

The need for treating Weapon System software as a contract deliverable and a designated CI has been widely recognized. A number of previous studies (e.g., Tactical Computer Software Acquisition and Maintenance Staff Study (Ref. 6) and Project Pacer Flash (Ref. 7)) have recommended that this be done as a matter of acquisition policy. Discussion with Systems Development Corporation (SDC), TRW, and Boeing brought out the same recommendation. A similar point was emphasized by Fleet Combat Direction Systems Support Activity (FCDSSA). It is now USAF policy in AFR-800-14 (Ref. 2).

As a result of this recognition, most major system contracts now require operational software as deliverables (although there have been notable exceptions). However, practices vary with respect to other classes of software. In a system development of any size, computer program developments may include:

1. Tactical on-line (operational) programs
2. On-line equipment test and fault isolation programs (Automatic Test Equipment, ATE)
3. Development support programs, including
  - a. Compiler/operating system and other program generation programs
  - b. Subprogram test drivers
  - c. Simulation-driver programs for computer program and system evaluation
  - d. Equipment checkout assist programs
4. Data collection reduction and analysis programs
5. Supporting simulation and integration programs.

Support Software. In the case of the B-1 avionics program, the support software as well as the operational program, is a deliverable. Since the support system was part of the Boeing software support facility and was in considerable measure developed on company funds, the delivered system is constrained to be used by the Air Force only for B-1 support. This case highlights a general complication with making support software a deliverable. Special attention should be directed on policy in this area.

In some developments such as Trident and P-3, the compiler has been a Government-furnished item. Few systems have specifically called out simulations and test-driver programs as deliverables, even though these may be necessary for life support of the operational programs.

Documentation Requirements. One problem in the designation of deliverable computer programs is the required documentation effort. Full formal documentation, as called out in MIL-STD-490 (Ref. 8) and detailed in documents such as SECNAVINST 3560.1 (Ref. 33), is extensive and costly. There are a few guidelines for definition of a lesser documentation set for support programs. Pershing is an example of selective documentation, mainly driven by cost considerations. In that system only two of over 149 programs are documented and controlled to full MIL-STD requirements. The remaining programs have documentation tailored to their natures and applications. This point is further discussed under Subsection 6.6.2, Software Acquisition Guides, AM2.

In computer program developments directly managed by FCDSSA, a somewhat different approach to computer program delivery has been used. For these systems FCDSSA has used in-house level-of-effort contracting, in which the deliverables are phased steps in development, defined by task statements. In this system FCDSSA itself retains responsibility for design and performance, rather than the contractor.

#### Implementation

Provide clear guidelines for designating appropriate computer system resources (computer programs and computer data) as CI's in the Program Management Directives and manuals. Call for scheduled delivery, like hardware items. Specify support and test and integration software as separate deliverables.

## 6.2 ACQUISITION PLANNING

### 6.2.1 MILESTONED DEVELOPMENT PLAN

AP1

#### Recommendation

Define the requirements for milestones in the Full Scale Development phase to ensure the proper sequence of analysis, design, implementation, integration, test, and review processes. Also, define criteria that will be used to demonstrate that each milestone has been achieved.

#### Primary Problem Areas Addressed

- Lack of Policy Guidance and Planning
- Inadequate Management Procedures
- Unrealistic Cost and Schedule Estimates
- Premature Programming

#### Discussion

An essential element of effective management of a major development program is a well-defined set of milestones and deliverables that the contractor and program manager can use to guide and assess the orderly progress of the work. Because software is inherently abstract, it is particularly necessary that a milestone development plan be used in its acquisition management. Systems that identified milestone planning as a major management technique include DLGN-38, AEGIS, CV, S-3A, Trident, Pershing, and SAM-D.

Weapon Systems Milestones. The actual milestone definitions and level of design control involved varies from system to system. In DLGN-38 management, the many tasks required in development of the total ship combat system are defined with schedules and deliverables in an Integrated Combat System Management Plan (ICSMP, Ref. 29). This document is used as a parent for more detailed subsidiary plans.

The AEGIS system of milestones identifies major decision points in a multiphase development program. The milestones are keyed to major design reviews and system tests, and are supported by identified design documents and reports.

Trident has used a system of phased, milestone deliveries as a more detailed management and technical tool. These deliveries are identified with steps of program coding completion in a top-down design and implementation approach. The milestone delivery points, coupled with Software Status Reports, are used to monitor development progress, to manage integration and testing, and to identify, as early as possible, interface problems among subsystem programs and between the subsystem programs and the operating system.

SSD 61-47B Milestones. While there are numerous management and documentation standards which refer to milestones, none really clearly define a set generally suited to software acquisition, except Air Force SSD

Exhibit 61-47B (Ref. 3). While this document is 9 years old, and supposedly was made obsolete by MIL-STD-483 (Ref. 9), it is still used as a basic reference. SECNAVINST 3560.1 (Ref. 33) defines a series of requirement, design, and test documents and gives a flow chart showing the general order in which they would be produced during a system development. However, while the documents are well defined, they do not appear to cover the steps normally taken during acquisition as clearly as does SSD 61-47B.

While SAM-D, AEGIS, Trident, S-3A, CV, and Pershing established their own milestones, the B-1 avionics program obtained concurrence from the Air Force to base its contract on the SSD 61-47B milestones. Figure 6-2 shows how the milestones correlate with the Part I and Part II items in MIL-STD-483. It is seen that Milestone 2 encompasses some of both Part I and Part II requirements.

As a matter of interest, the milestones thus defined are listed in Table 6-2.

The SSD 61-47B milestones are directly related to the Preliminary Design Review (PDR) and the Critical Design Review (CDR) prescribed for major development programs. PDR occurs after Milestone 2 and CDR after Milestone 4.

#### Implementation

Amplify the definition of requirements for Preliminary Design Review (PDR) and Critical Design Review (CDR) to specify the items of analysis, design, implementation, integration, and testing to be completed. Develop an updated version of the milestone definition of SSD 61-47B (Ref. 3) or its equivalent. (Note that Milestone 1 of SSD 61-47B should precede Full Scale Development.) Incorporate these in the Program Management Plan and specify that the milestone provisions be written into development contracts.

Milestone* Document	M/S 1	M/S 2	M/S 3	M/S 4	M/S 5	M/S 6
MIL-STD-483						
Part I						
Interface Req	•	•				
Functional Req	•	•				
Equip and Facility Req	•	•				
Special Req	•					
Quality Assurance Req	•					
Part II						
Functional Allocation		•		•		
Storage Allocation		•		•	•	
Data Base Characteristics		•	•			
Functional Flow		•	•			
Quality Assurance Provisions				•	•	•
Module Description		•		•	•	

\* Milestone Pattern After SSD Exhibit 61-47B

Fig. 6-2 MIL-STD-483/Milestone Document Correlation Matrix

TABLE 6-2  
SOFTWARE MILESTONE DOCUMENT DEFINITION

Milestone	Document	Content
1	Software System Design Criteria	Software System Design Requirements <ul style="list-style-type: none"> <li>• Functional requirements</li> <li>• Particular accuracies, logic, mathematics, or constraints</li> <li>• Interface requirements</li> <li>• Facility and equipment requirements</li> <li>• Test requirements</li> </ul>
2	Implementation Concept and Test Plan	Software System Preliminary Design <ul style="list-style-type: none"> <li>• Functional allocation</li> <li>• Software module description</li> <li>• Interface and data base</li> <li>• Mathematics and unique logic</li> <li>• Test plan</li> </ul>
3	Software System Interface Specifications	Interface Requirements between Operational Functions <ul style="list-style-type: none"> <li>• Define software system environment</li> <li>• Data base and control interface data</li> <li>• Software system data flow definition</li> <li>• Operational functional flow definition</li> </ul>
4	Software Module Design Specifications	Software Module Detailed Design Requirements <ul style="list-style-type: none"> <li>• Detailed design requirements for each module using approved M/S2 and M/S3 documentation as a base</li> <li>• Software module acceptance test procedures and success criteria</li> </ul>
5	Software Module Documentation	Release of "as built" Documentation <ul style="list-style-type: none"> <li>• Updated M/S4</li> <li>• Storage requirements</li> <li>• Critical time dependencies</li> <li>• Test card decks and listings</li> </ul>
6	System Test and Acceptance Specifications	Software System Level Test Plan (Category II) <ul style="list-style-type: none"> <li>• Overall testing concept</li> <li>• Test conditions and purpose</li> <li>• Test equipment requirements</li> <li>• Detailed acceptance test criteria</li> <li>• Detailed test procedures</li> <li>• Test data listings and decks</li> </ul>
7	Operating Instructions	All the information that is needed to use the computer software subsystem in the operational modes
8	System Configuration Index	Summary Description of the Deliverable Software Subsystem

6.2.2 COMPUTER SYSTEM RESOURCE DEVELOPMENT PLAN

AP2

Recommendation

Ensure provision of a detailed Computer System Resource Development Plan as part of the bid package on Full Scale Development contracts. The plan should cover all aspects of the contractor's approach to organization, design, test, management, documentation, and other aspects of the program.

Primary Problem Areas Addressed

Inadequate Management Procedures  
Lack of Established Standards  
Unspecified Support Requirements

Discussion

A comprehensive Computer Systems Resource Development Plan is needed for several essential purposes. It ensures that all aspects of the development of major software systems are well defined and understood, and that the management organization is appropriate for the task. It provides a mutually agreed-upon plan for engineering management. It also provides a source of information concerning a contractor's experience, understanding, organization, and facilities that is invaluable in making an informed selection among competitors.

AFR-800-14, Vol. II (Ref. 10), calls out the requirements for a Computer Program Development Plan (CPDP). The CPDP identifies the actions needed to develop and deliver computer program configuration items and necessary support resources. It is to be prepared by the implementing command or, if the development effort is contracted, the plan may be prepared by the contractor and approved by the implementing command. The CPDP addresses the following items:

1. The organization, responsibilities, and structure of the group(s) that will be designing, producing, and testing all computer programs.
2. The management and technical controls that will be used during the development, including controls for ensuring that all performance and design requirements have been implemented.
3. The methodology for ensuring satisfactory design and testing, including quality assurance.
4. The development schedule for each computer program configuration item and proposed program milestone review points.
5. The procedure for monitoring and reporting the status of computer program development.
6. The resources required to support the development and test of computer programs. Special simulation, data reduction, or utility tools that are planned for use in the development of computer programs should be identified.
7. The general procedures for reporting, monitoring, and resolving computer program errors and deficiencies during development and testing.
8. The methods and procedures for collecting, analyzing, monitoring, and reporting on the timing of time-critical computer programs.



9. The management of computer program development masters, data bases, and associated documentation including its relationship to the Configuration Management Plan.
10. Guidelines and checkpoints for ensuring future computer program growth, modularity, and ease of modification.
11. The approach for developing computer program documentation.
12. Training requirements and associated equipment for the deployment phase.
13. Engineering practices to include: standards, conventions, procedures, and rules for program design; program structures and conventions; display and logic standards; input/output signal standards; and other disciplines affecting development.
14. Security controls and requirements.
15. Simulation techniques and tasks.

Examples of the Computer Program Development Plan can be found in AEGIS and in Trident. They form the basis for resource control and for detailed design program monitoring. Discussions at Systems Development Corporation (SDC) stressed the crucial importance of such a plan in any major computer system acquisition program.

#### Implementation

Require that the Program Manager prepare a set of development requirements to be included in the Request for Proposal (RFP). Specify those aspects that are directed by the Government. Specify the nature and scope of description required in the contractor's Computer System Resource Development Plan.

### 6.3 SYSTEMS ENGINEERING

#### 6.3.1 SYSTEMS ENGINEERING OF COMPUTER SYSTEMS

SEL

##### Recommendation

For systems involving several distinct functions, require that the system be divided into functional segments in accordance with the operational requirements. Require during the Program Validation Phase that tradeoff analyses be performed for hardware versus software (i.e., hardwired versus programmable functions) and for different computer system architectures.

##### Primary Problem Areas Addressed

- Lack of Software Systems Engineering
- Inefficient Processing Architecture
- Inadequate Hardware/Software Tradeoffs
- Nonmodular Software Architecture

##### Discussion

The discussion of the analysis of system requirements under Recommendation MP1 noted the stage in which the overall system configuration was developed in conjunction with the translation of system requirements into an organization of major hardware/software elements. This involves several important decisions.

The decision as to the functions that are to be performed by special purpose hardware, as opposed to programmable general purpose computers, is one of growing importance. Past practices have tended to assign the central computer every type of data processing, with the result of stressing or exceeding its time budget and memory capacity. Integrated circuit developments have recently made it possible to build inexpensive special purpose digital hardware that can do a great deal of high speed processing, such as sensor data handling very economically. Still more recently the LSI technology has produced microprocessors that are programmable and inexpensive. In either case it is evident that modern systems should seek to relieve the central processors of as much high speed repetitive data processing as practicable in the interest of overall performance and cost.

Given the functions to be performed in computers, another major decision is the degree of centralizing combat system processing, i.e., whether a few large processors or a larger number of small dedicated processors should be used. In the developmental stages of the Naval Tactical Data System (NTDS), all digital processing was combined in an integrated computer complex. Today, the automation of sensor processing and the introduction of digital fire control have led to a more distributed architecture, with centralization confined largely to the combat direction or C&C system.

Functional Segmentation in Weapon Systems. The DLGN-38 computer programs have been developed as five separate software systems. Two of these, the Command and Control System (C&CS) and Sensor Interface Data System (SIDS), operate within a shared-memory AN/UJK-7 computer complex, using an extended memory area as a controlled system interface.

AEGIS, developed under a single prime contractor, segmented the major system elements according to function, with major software developments

in the AN/SPY-1 radar, Command and Decision, Weapon Control, and Operational Readiness Test Systems. The Tactical Executive Program, which is used in common by each system element, was also treated as a configuration item and a separately controlled development. Program interfaces with the Executive are treated in the Computer Program Interface Document (CPID) in basically the same manner as are interfaces between systems.

Segmentation has been especially beneficial in the AN/SPY-1 radar. Design responsibility for both radar equipment and control computer programs was placed in the hands of one manager. This allowed extensive hardware/software tradeoffs to proceed in a disciplined fashion, without undue impact on other elements of the system.

A hardware configuration analysis has been recognized as a significant task in a number of other systems. Prior to DSARC II, Trident conducted an in-depth, detailed Proposed Technical Approach (PTA) study that examined a number of centralized and decentralized systems. The PTA reduced the viable alternatives to four. The Navy Ship Acquisition Programs Manager (SHAPM) (with PM-2 approval) then selected a centralized computer system concept.

Extensive system tradeoff analyses were conducted by Raytheon during the Engineering Development Definition Phase of SAM-D. The analyses included a study to determine the system functions allocated to the dedicated hardware of the radar unit and to the general purpose computer in the weapon control unit.

Interface Management. The success of system development rests in large measure on the effectiveness with which configuration management is performed. This in turn depends on the relative simplicity of the interfaces, which further depends on how closely the physical segmentation corresponds to functional independence, and on how clearly the interfaces are defined at the outset of the program.

The degree of complexity involved in major systems acquisitions often leads to overlapping and/or redundancy of functions among one or more segments of the total system unless strong measures are taken at the beginning of the program to ensure an understanding of the basic purpose of each segment. Failure to provide this definition leads to confusion of design and loss of configuration control, which inevitably lead to substantial rework and impaired final system performance.

In Navy documentation requirements (SECNAVINST 3560.1, Ref. 33) it is now recognized that the need exists for top-level system requirements and design documentation, and for interface design specifications that establish in detail the relationships among system elements. However, the engineering design process, both in-house by a single contractor and between two or more

contractors, is such that differing interpretations of requirements and specifications become the norm rather than the exception. A means must be established to ensure correct design communications between the various individuals and agencies involved. Failure to do this leads to a significant level of redesign at a later date and a consequent loss of money and time.

Implementation

Call for these tradeoff analyses to be performed during the Program Validation Phase and presented at System Design Review (SDR) and DSARC II.

### 6.3.2 PROVISIONS FOR GROWTH IN SYSTEM REQUIREMENTS

SE2

#### Recommendation

Provide for growth and change in requirements on Weapon System computer software by identifying parameters that are uncertain or are likely to change in the future and, where possible, specify the probable limits on such changes. Also identify novel environments and use of new techniques. Require that computer systems be sized to provide for uncertainties and requirement growth.

#### Primary Problem Areas Addressed

- Lack of Requirements for Adaptability
- Inadequate Processor Capacity
- No Provision for Changing Requirements

#### Discussion

The inherently complex, uncertain, and growing threat environment in which modern Weapon Systems operate ensures that changes will be needed in system requirements during its lifetime. It is a basic feature of software that it can accommodate change provided it is not limited by hardware capacity or speed. Accordingly, an important part of software system engineering is the judicious and controlled provision of growth capability. In developing a new system, still further margin must be provided to accommodate the inevitable additional processing required to handle factors not foreseen during preliminary design.

The consequences of changes and growth in system requirements are twofold: (1) modification of the operational program within the constraints of the processor, and (2) expansion of processor capacity. With regard to the first consequence, the main impact is that of cost. Costs of modification can become enormous if the processor is operating near its capacity and extensive repackaging of the program is required to make room for the modification. This argues strongly for the provision of space capacity at system delivery as the Navy has done in TADSTAND-5 (Ref. 14). The problem is that the amount of space capacity that should and can be provided varies with many factors, including the maturity of the program and space and weight constraints on platforms such as aircraft.

There is a factor of the modification costs that can be affected by careful design planning and this has to do with the manner in which the program is designed to accept change. This is discussed further under Recommendation SE3. The identification of parameters subject to change can also be of great importance in providing a basis for allocating space capacity and designing for minimum impact.

The Electronics-X study (Ref. 12) and the report of the Army Scientific Advisory Panel (Ref. 13) both make note of the problem of overburdened processors and recommend special attention to the adequate sizing of computer hardware.

Experience from Weapon Systems. Nearly all the Weapon Systems investigated have experienced a problem with growth in the demand for computer system resources. The early Naval Tactical Data Systems (NTDS), which used the AN/USQ-20 series of computers, were especially constrained in available memory because of the state of technology and memory costs at the time these systems were designed. Various techniques were employed to constrain use of memory or to overlay program capabilities from peripheral memory. However, some desired functions were not implemented because of memory limitations. In the NTDS Model IV development now in progress, the older systems will be supplemented with an extended memory unit (262k 32-bit words), which will relieve many of the long-standing constraints.

Changes are particularly probable in the Command, Control, and Communications functions of the systems because CC&C must adapt to changing tactics and environment. Computer systems with specific technical tasks, such as fire control systems, are less likely to experience growth after initial development, except as a phased capacity expansion.

Aircraft and space systems have also been memory constrained, having peculiar space, weight, and power limitations. Several Navy aircraft systems have made use of disk or drum memory to extend functional capability. The more recent systems, based on the AN/UYK-7 or similar computers (as in SAM-D) have an inherent potential for expanding computer resources because of the modular design of the computer hardware.

An initial reasonable provision for growth has not always proved adequate. For example, the E-2C System recognized the necessity for substantial growth margin and provided a second processor, representing 100% margin over initial estimates. This was expected to easily absorb anticipated growth and computer overload caused by hardware design uncertainties. However, largely because of increased radar data processing requirements, all of the margin was used. A software tailoring effort was required to remain within available computer resources.

Approaches to Growth Problem. A number of approaches to the growth problem can be adopted. The most important of these is a detailed analysis and definition of system requirements as discussed in Recommendation MP1. An adequate analysis of this type, undertaken early in development, should reduce the surprise factor in computer growth. A disciplined hardware/software tradeoff, as suggested in Recommendation SE1, should also force detailed examination of true computer resource requirements. In addition, a plan for orderly growth, which is the subject of this recommendation, is also needed as an element of the systems design requirements.

The consensus of the Weapon System community, the services, and industry is that the need to provide for growth in system requirements is clearly great. Highlighting provisions for growth in requirements will not only assist in meeting mission objectives but will also extend the life and greatly improve the flexibility of the operational system.

Implementation

Require (in appropriate regulations) that the design requirements or specifications that are given to the software developer contain parameter uncertainty limits. Specify provisions to accommodate parameter changes in the Preliminary Design Review (PDR) process. Provide for software breadboarding (including cost and scheduling) in novel environments or when using new techniques. Provide in the requirements that substantial reserves of time and memory be delivered; where possible they should be tentatively allocated to specific potential growth requirements. Check all these items at the PDR and at the DSARC II and III reviews.

### 6.3.3 SYSTEMS ENGINEERING OF COMPUTER SOFTWARE

SE3

#### Recommendations

Specify the use of modular software architecture and an orderly phased design approach for developing major computer programs that defines the higher levels of the program and then progresses to design and test successively lower levels. The latter approach is often referred to as top-down design. It involves the formal definition of a hierarchy of program elements and restrictions concerning lateral communications.

#### Primary Problem Areas Addressed

- Non-Modular Software Architecture
- Bottom-Up Design
- Premature Programming

#### Discussion

The application of a systems engineering approach to computer program design requires that each operational program be subdivided into a set of functionally distinct subprograms with simple and cleanly defined interfaces. Further application of this principle dictates that the architecture of the total program should consist of a systematically organized hierarchy of operational blocks, with strictly limited lateral communications.

Modular Design. Modular design of computer programs is not a new concept. It has been employed in Naval and other computer systems for many years. The Fleet Combat Direction Systems Support Activity (FCDSSA) approach has moved toward standardization of module design and currently uses common software modules among ships of the same class (in which the computer program implementation may vary owing to differences in installed equipment).

While the value of modular design is generally appreciated, it is often not enforced as an engineering discipline, or when it is, only in a nominal way. It is proposed that strong emphasis be placed on program organization during the early design stage so as to achieve the advantages of simple and well defined interfaces resulting from good functional program structuring. In particular, such organization should take into account functions that are most likely to change during or after development, and separate them from those likely to remain the same. This has proved to be highly effective in the Site Defense Program in minimizing the cost and schedule impact of changes in requirements. Similarly, segregation of input/output processing is advantageous for maintaining maximum flexibility.

Another valuable feature of modular design is that it can lead to a set of traceable requirements, beginning with performance specifications and ending with production design specifications. If the organization of the program is not established during the definition phase, there is almost inevitably a lack of any direct part-by-part relationship between the successive phases of design, with the consequent difficulties in evaluating whether the design meets requirements.



Top-Down Development. The orderly phased approach to program design, referred to as top-down, is relatively recent as a formalized technique. However, its principles are quite basic and have long been in general use as a systems engineering discipline. It simply means that before a program component is built, it should be designed, and before it is designed, it should be defined as part of the larger whole. The documentation system defined in MIL-STD-490 (Ref. 8) and further expanded in SECNAVINST 3560.1 (Ref. 33) implies a structured top-down definition and allocation of design requirements. The potential benefits of top-down development are improved program definition and more effective product integration. The approach is more likely to yield a well considered hierarchical design with clear definition of function and modularity.

Among the previous software management studies, Electronics-X (Ref. 12) recommended "completing the design of the system and the basic program structure in substantial detail before making major commitments to hardware or coding." CCIP-85 (Ref. 22) urged the use of methods "bringing structure to the programming process, ranging from establishment of extensive program quality standards to more sophisticated techniques of software engineering."

It is sometimes implied that top-down means that the complete system is to be designed before any of it can be implemented. This is not what actually occurs in practice. It has been stated earlier that preliminary design should include not only modeling but actual implementation of critical parts of the program. This is necessary to ensure that a firm basis exists for allocating time and core to each program module, and hence, that the initial design will be stable. However, while some portions of the program may be implemented before others are designed, any given portion is developed top-down. Further, the hierarchical architecture and definition of interfaces ensures that the individual portions are sufficiently independent that they may be developed separately.

Top-Down versus Bottom-Up Development. The difference between top-down and the traditional bottom-up software development may be illustrated with reference to Fig. 6-3, which has been used in many papers on this subject. While this diagram is somewhat idealized (in practice, the program hierarchy is far from binary), it suffices to make the comparison. In both the upper and lower portions of the diagram, the program development has been approximately half completed. In the upper portion, illustrating bottom-up development, one of the modules is shown to be completely built while the other module is still being constructed.

In the bottom-up approach, special test drivers, as indicated in the figure, must be designed and built for each successive level of integration and testing. Four test drivers are shown in the diagram. None of the program can be "run" with a realistic simulator until all elements are completed and assembled.

The top-down process, illustrated in the lower portion of Fig. 6-3 is quite different in this regard. The only necessary test input is seen to be a simulator exercising the total program. The top block corresponds to

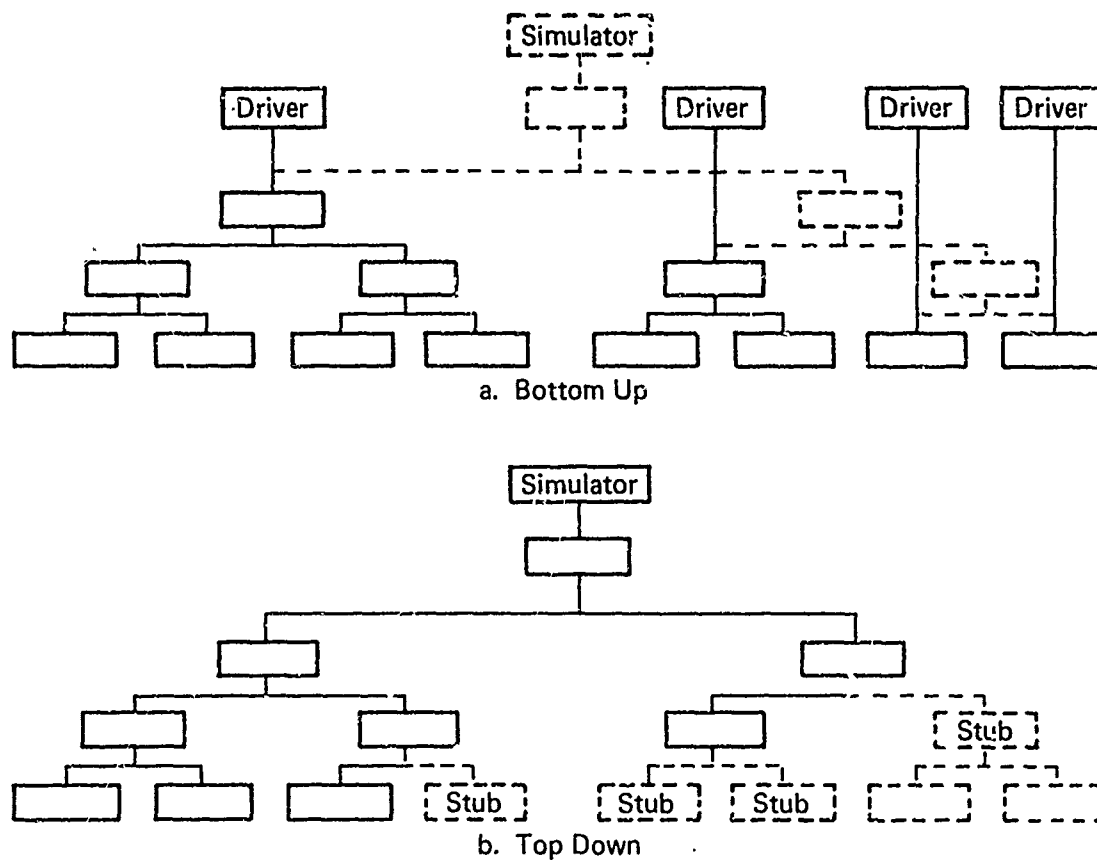


Fig. 6-3 Bottom-Up Versus Top-Down Software Development

the program control structure, and it is shown to be completely designed and implemented. The diagram shows one of the modules implemented except for one routine, which has been dummied with what is called a "stub." The second module is being designed at present with only the first few levels implemented; the missing routines are present only as stubs. Actually, individual routines should be tested prior to assembly in either approach in order to remove most errors at the earliest time. However, when the control structure is developed first, each block can be added as soon as it is implemented and checked out. Thus the program can be run early in the process using dummy modules in place of those not yet designed and built. This not only simplifies the testing process but also uncovers errors early in the design cycle.

Applications in Weapon Systems. The Trident Command and Control System was the first Navy system to formally specify use of top-down design. Its use was addressed in the Command and Control Software Management Plan. The requirements included not only the application of the procedure to the overall system, but also to the major modules of the system. SAM-D used a top-down approach to software design but a conventional bottom-up approach to coding.

AEGIS employed a top-down approach to design, implemented through a structured series of development specifications and supplemented by controlled interface documentation and by Functional Flow Diagrams and Descriptions (F2D2) developed by the systems contractor, RCA. The software contractor, Computer Systems Corporation, also used the "Threads" technique for function tracing.

#### Implementation

Specify the use of modular top-down design in the Request for Proposal (RFP) and use as a criterion in examining the contractor's definition of his organization of design, production and test, design methodology, and engineering practices.

#### 6.4 IMPLEMENTATION PROCEDURES

##### 6.4.1 SOFTWARE DEVELOPMENT SUPPORT TOOLS AND FACILITIES

IP1

##### Recommendations

Ensure that the Full Scale Development program includes provision of adequate modern support tools and facilities, including such items as assemblers, compilers, editors, debug aids, data base and library management systems, and associated operating systems. Require maximum use of existing proven tools and facilities. Provide that any of these tools and facilities that will be required by the Operational Support Agent for system maintenance be delivered in transferable form and also be capable of application to future Weapon System programs.

##### Primary Problem Areas Addressed

Long Testing Due to Errors and Changes  
Support Systems Not Established

##### Discussion

Development Support Software is a major cost item in Weapon System Software Acquisition for two reasons. First, the size of support programs may be two to five times larger than the tactical software. If it is not available from previous programs and must be developed along with the tactical software, it becomes a large fraction of the total program cost as well as a schedule pacing item. Secondly, if a major software development is attempted with only rudimentary support software, the time and cost of development, especially of the test and verification phase, can become excessive. It is clear, therefore, that management attention to the availability of adequate support tools is most important. Serious problems can arise if development support tools and facilities are underdeveloped or inadequate to the development in progress.

A major software problem encountered by AEGIS was the incomplete status of the compiler (CMS-2) at the start of the system development, which caused added costs and delays.

Parallel development of the unique SAM-D computer and SAM-D software greatly complicated software development and checkout. The SAM-D Program Manager's office also described difficulties in checkout and monitoring of software development because neither the computer nor the compiler for the high-level language selected for the system was fully developed at the start of software development.

Industry Software Support Tools. Support software includes compilers, assemblers, editors, data base and library management systems, linkers, simulators, and related tools. It can also cover a spectrum of design, test, documentation, management information, and system qualification tools. Most major software system contractors have highly developed support tools, and perform most design and test operations on a powerful host computer facility. An example of the application of a major support facility is shown in Fig. 6-4 taken from a Boeing presentation. They anticipate substantial cost improvements through future development of algorithm banks and software design automation.

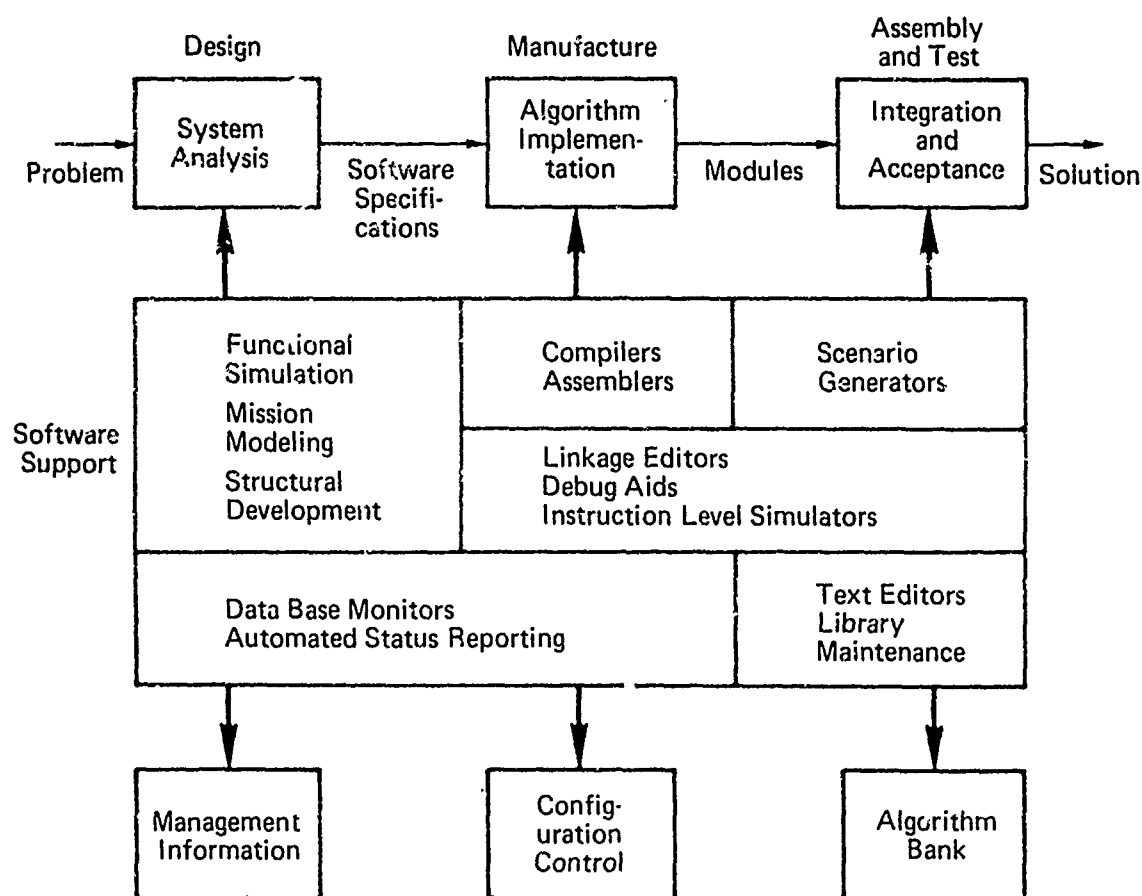


Fig. 6-4 Support Software in Software Engineering

Figure 6-5 indicates the range of specific development, test, and documentation tools used by TRW on its 200k instruction Space Vehicle Dynamic Simulation program. The special tools shown on the right have been developed by TRW for use on all major programs. Other major software contractors are also developing powerful arrays of tools. Several of the software contractors now have code-checking tools that check every line of code in the system. Documentation tools exist that are capable of automatic flowcharting and sub-routine textual description. These aids can also be invaluable to the software maintenance agency, which in the past has frequently been plagued with documentation that was incomplete and did not match the code. The trend is now toward integrated tools that cover a broad spectrum such as the Systems Development Corporation's (SDC) software factory and CSC's Threads which can address the management as well as the technical aspects of software acquisition.

Support Tools in Weapon System Acquisition. Several earlier studies discuss the importance of software tools in Weapon System acquisition. The Army Scientific Advisory Panel Study (Ref. 13) urges "early development or selection of software testing tools." The Air Force CCIP-85 study (Ref. 22) recommends the use of test tools to validate software, thus "reducing significantly the danger that software errors could escalate crisis situations or degrade defenses at critical times." Project Pacer Flash (Ref. 7) recommends the "Establishment of an automatic test equipment capability within AFSC divisions to assure the application of the automatic test system to Weapon Systems during DT&E and production."

FCDSSA(SD) uses extensive support tools and facilities and periodically updates the support software. The most recent operating system, designated SHARE-7, operates in a UYK-7 computer and contains compilers, host-computer emulators, and debug aids. SHARE-7 is also used during the first phase of software validation and integration development.

The AEGIS System uses major Program Generation Centers at the software subcontractor's facilities. These centers are independent of integration test activity. They are provided for in the Computer System Development Plan.

#### Implementation

Specify support tool and facility requirements in the Request for Proposal (RFP) and evaluate proposals on the adequacy of existing and planned support facilities and tools, and on the contractor's experience in their use. Have the Operational Support Agent participate in defining the requirements. Define support software as a Configuration Item. Provide that the portion of support software needed for operational maintenance be a contract deliverable with formal documentation. Make provision for support planning in Acquisition Management regulations, and subject it to design review procedures. Provide O&M funds for measures to ensure transferability. Provide for development of new tools in a manner directly transferable to other programs. Continue support and provide funding for the DoD Software Management Steering Committee panel's work on a software catalogue.

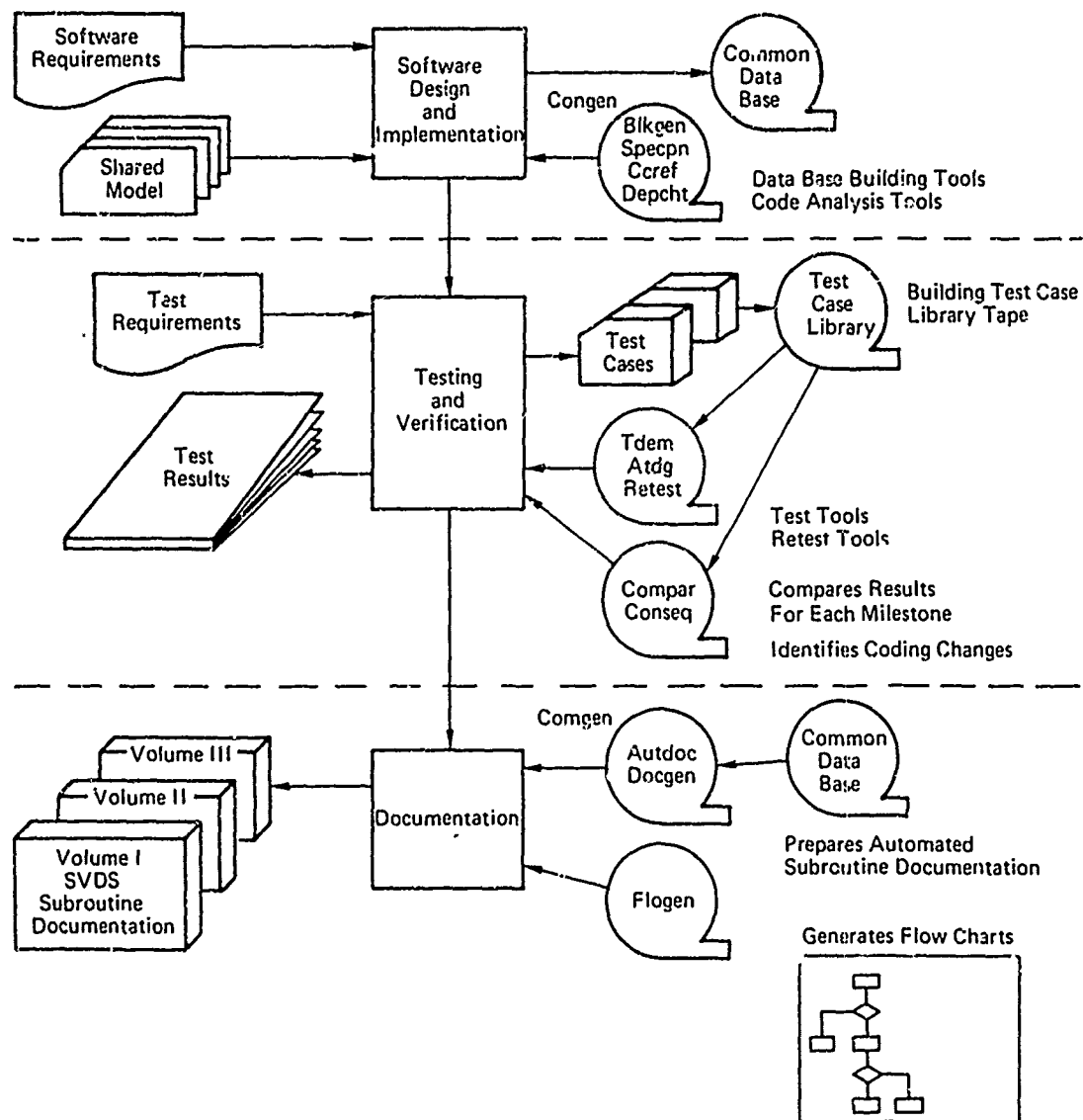


Fig. 6-5 TRW/JSC Program Utilizes Software Development Tools for all Phases of Development

#### 6.4.2 DISCIPLINED PROGRAMMING

IP2

##### Recommendation

Require that the computer program development contractor apply a highly disciplined set of engineering practices to the detailed design and programming phases of development. This must involve a clear and disciplined set of standards covering program structure, size, control, interface, formal conventions on data base management, and the demonstration that the standards are enforced in practice.

##### Primary Problem Areas Addressed

Lack of Established Standards  
Unstructured Programming

##### Discussion

Programming is a term that is used to refer to the process of detailed software design as well as to the subsequent steps of implementation leading to a coded and checked out program. Software design may be viewed as the process of translating the computer software requirements into basic functions with a common data base and defining logic, interfaces, control flow, and timing. Implementation is the actual coding of the program in some computer language including the debug and check-out steps.

In discussing the subject of disciplined programming it is important to distinguish between the design and implementation phases, because the two processes often mesh intimately and are sometimes confused and mistakenly lumped together. This practice can lead to the common phenomenon of "undesigned software", i.e., software that is merely implemented. This current section concerns disciplined programming of implementation, whereas the subject of software design was treated under 6.3.3.

The lack of established programming standards consistently leads to unstructured and premature programming. This, in turn, results in low unit productivity, inadequate interface management, and long, costly testing because of errors and changes. Integration, verification, and validation tend to be very costly because of the lack of early attention to interface problems and overall system requirements.

Undisciplined and unstructured implementation also tends to limit the utility of design reviews. The final product frequently ends up being a conglomerate of program patches that are very difficult to maintain. Different assumptions made at bottom-level elements often result in higher level data and control structures merely strung together to operate in the overall configuration. Such an amorphous structure is cumbersome to update in order to meet changing requirements. Weapon Systems development is replete with instances of software that is poorly documented and hard to maintain.



### Elements of Disciplined Programming

Several studies and workshops address the issue of programming discipline and practices. Electronics-X (Ref. 12) urges the investigation of "methods of bringing structure to the programming process, ranging from establishment of extensive program quality standards to more sophisticated techniques of software engineering." In his paper to the Aeronautical Systems Software Workshop (Ref. 25), W. L. Trainor highlights the following areas that software production standards should address in order to achieve the goal of "quality" software:

1. Coding conventions, such as use of indentations, spaces, etc.
2. Documentation conventions, such as use of comments within program listings to improve ease of comprehension.
3. Labeling and naming conventions and restrictions to produce consistent terminology.
4. Instruction-use conventions and restrictions.
5. Conventions for parameterization and reuse of modules.
6. Conventions for assigning attributes to data and constants.
7. Input/output conventions and restrictions.

Fortunately, industry is developing new techniques and standards in this critical area. TRW imposes rigorous programming discipline on its Site Defense Software development. The length of any individual routine is limited to 100 Fortran statements, and subroutine re-entry is tightly controlled. A Unit Development Folder is maintained for each routine, which gives the status, schedule, test cases, code listing, and other information pertinent to the program. An automated tool checks the code for compliance with established programming standards.

### Structured Programming

IBM recommended the use of the Structured Programming technique which has been important to Trident program development. The technique involves maintaining a program organization discipline in which only certain basic control structures are allowed. It has been shown that any application program can be written using three basic structures, notably: DO WHILE, IF THEN ELSE, and SEQUENCE. However, additional structures such as DO UNTIL and DO CASE are helpful to provide more convenient implementation.

This technique has advantages during implementation, testing, and maintenance. It is stated to have certain disadvantages, such as loss of common or reusable components and inefficiency in handling asynchronous control and error exits, and there has been some resistance to this particular discipline for real-time applications. However, as indicated, Trident is using it and AEGIS also plans to implement Structured Programming in its next development phase.

What is important, at this point, is not so much which specific programming discipline, such as Structured Programming, is selected, but rather that programming be controlled by one means or another. The Request

THE JOHNS HOPKINS UNIVERSITY  
APPLIED PHYSICS LABORATORY  
LAUREL, MARYLAND

for Proposal (RFP) should call for the contractor's programming procedures, and their evaluation should be an important criterion in the selection of the software contractor.

Implementation

The RFP should call for a description of the contractor's design and coding manuals and his approach to programming discipline in the Computer System Resource Development Plan. Formal and well-established procedures that have been demonstrated on prior programs should be an important element in the contractor selection process. The contract should specify that the proposed procedures be used.

#### 6.4.3 SYSTEM INTEGRATION AND TEST CAPABILITY

IP3

##### Recommendation

Require that an integration and test capability be provided as part of Full Scale Development of major Weapon System software. This should be a software test bed combining simulated elements and real hardware (including operator consoles) to be used in progressive integration and test of system elements. It should provide real-time dynamic stimuli and responses under repeatable and off-nominal test conditions. The portion of this capability that is required for Operational Support and Maintenance should be specified to be transferable or capable of duplication.

##### Primary Problem Areas Addressed

Long Testing Due to Errors and Changes  
Maintenance Provisions Not Included

##### Discussion

The section on Software Development Support Tools and Facilities (IP1) discussed delivery of the tools associated with the support of programming and the testing of individual program elements. In order to test an operational program with inputs representative of the operational environment, another class of test tools and facilities is required that simulates the real-time information inputs and interfaces with sensor and weapon requirements, as well as with other computer programs operating in the overall system.

It is necessary to consider at least three specific phases where simulation integration facilities are needed. The first is in support of system development, the second in support of system integration, and the third in support of system maintenance. The same facility may evolve through the development into the maintenance phase; more often the requirements are sufficiently different, or the need is simultaneous at two locations, so that two or even three different installations are required. In either instance, careful planning is necessary to ensure that adequate capabilities exist where needed, and at the same time, duplication is minimized.

Configuration. The general configuration of an integration and test facility is shown in Fig. 6-6. The operational processing system is shown in the center, and contains the computer program being exercised. The operational computer is connected by actual or simulated hardware interfaces to a simulation computer, and to such actual sensor or weapon equipment as may be available and appropriate to the stage of system integration. In the initial stages, there would be little or no real hardware involved. In the final stages, possibly at a support facility, a full complement of sensor equipment and weapon control equipment might be used. In the latter case, the facility may double for training purposes, as is often done in Navy systems.

An important part of a test or support facility is a set of operator consoles representative of the operational system. Since much of the task of Weapon System software is to provide displays for operator decision and control, evaluating this function early in the development is mandatory.

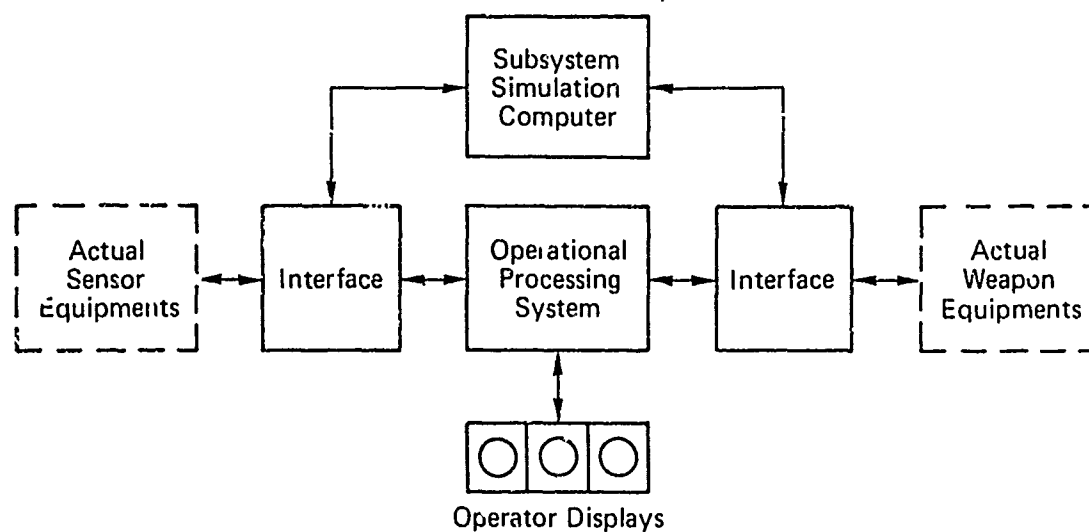


Fig. 6-6 Operational Computer System Integration and Test Facility

Another important function of the simulation software is the capability of not only imposing real-time dynamic stimuli, but also providing repeatable and off-nominal test conditions. Thus, the system can be stressed to the performance limits - a condition hard to impose with actual hardware inputs.

Avionic Systems Integration Facilities. All avionics programs reviewed had complete integration facilities, referred to as "hot-benches." Grumman Aircraft Corporation had an integration and test facility for the E-2C which contained all the actual hardware used in the system. It also had provisions for simulating many of the hardware interfaces. A similar facility was installed at FCDSSA(SD), which was the Operational Support Agent. The E-2C program has had a very good reliability record, having logged 1200 hours with no software errors. Grumman attributes a large part of this success to the extensive testing of the system.

A critical part of the P-3C acquisition process was the development and availability of the Integration Test Facility, which preceded coding. The early use of this to validate the code was considered essential. This test facility was made up of the major sensor, display, and computer interfacing equipment as well as the computer itself. Code was checked out in segments relating to individual subsystems and then brought together into a total system. In the subsequent P-3C update program, the Integration Test Laboratory at NADC, which is developing the program, is providing the same type of development support as the one previously used by Lockheed.

A comprehensive integration and test support facility was also developed for the S-3A. Program checkout and a phased sequence of integration steps were accomplished. The facility utilized actual and simulated equipment, and minimized the need for flight test to verify system performance.

At Boeing Aerospace Company, use of an Integration and Test Facility is standard operating procedure for all major Weapon System software development. Testing starts at the individual instruction level and culminates in the interfacing and dynamic exercising of the software by the operational avionics equipment.

In the case of avionic systems, simulation facilities have been required at the outset of program development to represent the flight dynamic environment. At the same time they have been sufficiently compact to make it practical to have them developed by the system contractor and used at his plant during program development. A similar system usually has been built for the Operational Support Agent. It is recommended that, especially in avionics systems, integration facilities be considered as a contract deliverable, making best use of the initial development to provide later for operational support. This does not mean that the support system should be the same as the development system, but that maximum commonality would be a major advantage as well as an economy.

Ship Systems Integration Facilities. For the ship systems reviewed, the nature of the integration facilities was quite different. Typically such facilities are at Naval establishments and are equipped with a substantial complement of combat sensors and Weapon Control Systems. Each of the escort ships investigated has been supported by a major facility at Mare Island. Facilities at FCDSSA have been used for subsequent Fleet support.

The FCDSSA's have extensive test and integration facilities. These provide capability for live mock-up of the system and software under test. There are provisions for non-real-time and real-time operation, peripheral simulation, operator or simulated operator input, and use of operational system hardware. Divisions within the FCDSSA Test Department provide test support, configuration control, delivery, and diagnostics.

Planning for AEGIS includes the concept of a Combat System Engineering Development Site (CSEDS), which is a major facility extending the Land-Based Test Facility concept to the Integration and Test needs of the total Ship Combat System. The CSEDS plan includes development of extensive simulators for system exercising, as well as inclusion of most major elements of the shipboard system.

Trident planned for and implemented a Land-Based Evaluation Facility (LBEF) during the conceptual phase of development. It has the specific purpose of software development, test, integration, and certification, and includes tactical equipment as well as computer systems. It supports total systems integration and verification of each tactical equipment suite prior to shipboard installation. It is located at the Naval Underwater System Center and used by the software contractor (IBM).

Planning and Control. It is recommended that planning for the Integration and Test Capability be done for the whole system life cycle and that the facilities built and the applicable support software produced during system development be made available for operational/maintenance support.

While the nature of the simulation software varies with the application, it is often larger than the operational program. Particular care should be exercised to control the scope of the simulation and test facilities to avoid overcomplication. This can come about either from attempting too much realism in the simulation, or requiring a full complement of hardware sensors and weapon control elements. Neither is really necessary for adequate testing, and both aspects should receive critical review during the evaluation of the technical plan, and subsequently at PDR and CDR, just as the operational software itself.

#### Implementation

Define the provision of an integration and test capability as a requirement in the Request for Proposal (RFP) and in the Computer System

Resource Development Plan. Specify that the portion of the simulation software required for system operational support and maintenance be made a contract deliverable with formal documentation. Provide O&M funds to the contractor for support of maintenance features. Constrain sophistication to avoid overcomplication, especially at the contractor facility. Make provisions for Integration and Test Facility planning in Acquisition Management regulations, and subject such planning to design review procedure. Consider training requirements for test facilities.

## 6.5 PROGRAM MANAGEMENT SUPPORT

### 6.5.1 TECHNICAL STAFFING OF PROGRAM MANAGER ORGANIZATION

MS1

#### Recommendation

Establish and implement a policy that Program Managers for major Weapon Systems be staffed with personnel experienced in systems engineering and software development and of sufficient stature and number to carry out essential management functions that cannot be delegated.

#### Primary Problem Areas Addressed

Lack of Policy Guidance and Planning

Inadequate Cost and Schedule Monitoring

#### Discussion

The success of software development efforts may be related to the level of attention and monitoring applied by the Program Manager's office. Previous studies have indicated a definite need for improving the staffing of the PMO's. The CCIP-85 study (Ref. 22) states that the services should: "Develop career paths and associated training programs and retention incentives for both commissioned and civilian personnel, allowing career advancement in technical disciplines associated with information processing."

M. R. Davis in his paper from Ref. 25, "Visibility and Responsibility in Aeronautical Systems Software," states that there are two basic questions to be asked: "What kinds of incentives are needed to induce the right kinds of people to come on board?" and "Is pooling of manpower resources a sensible interim solution until more people can be acquired?"

Weapon Systems Review Findings. Many of the Program Managers interviewed stressed the need for a qualified staff to oversee software developments. Also stressed was the shortage of personnel with adequate Weapon Systems software experience.

Two particular program offices which have been able to maintain strong control over software development are those for Pershing and AEGIS. The Pershing Program Manager has an engineering staff available to him at MICOM (Redstone Arsenal). AEGIS not only has computer program specialists directly on the Program Manager's staff, but has used software technical assistance from personnel in the office of the NAVSEA Technical Representative at the contractors' site, the technical advisor (APL/JHU), FCDSSA(DN), and other contracted advisors.

The SAM-D project office includes eight software specialists. This is considered to be a small staff. The Program Manager has contracted with IBM Federal Systems Division to provide an independent assessment of certain software efforts.



One Program Manager noted the difficulty of maintaining clear lines of responsibility and control over management tasks that are delegated to service agencies, although these agencies may have personnel with the needed expertise. In general, there is a shortage of software development know-how in the staff of responsible Program Managers. It is likely that many software acquisition problems would or could be averted if this staffing deficiency were improved.

Inherent Problems and Interim Solutions. A number of the military personnel interviewed noted the current difficulty of structuring a career specialty in tactical digital systems. Steps may be needed to recognize this specialty, to provide for career incentives, and to allocate experienced personnel where they are needed to assist development programs.

A fundamental difficulty, aside from career incentives, in finding qualified personnel in software engineering is that the educational system does not produce such individuals. A degree in computer science or training in conventional programming does not really provide the necessary background for Weapon Systems software engineering management. This is a nationwide problem and a solution to it at the service or DoD level is not apparent.

As an interim solution more consideration should be given to assignment of qualified software engineers from service laboratories to assist Program Managers. The experience acquired in such assignments would also provide the individual with an insight into Weapon System software acquisition problems which would be highly beneficial upon his return to laboratory programs.

#### Implementation

Provide for high level review (e.g., DSARC I and II) of Program Manager staffing at the start of Program Validation and the Full Scale Development Phases of major Weapon System development programs. Provide means for temporary assignment of engineers from service laboratories and support activities to fill key staff positions. Provide career incentives to attract competent engineers from within and from outside the Government into both military and civilian positions. Establish policies that assure adequate grade levels for Civil Service jobs in this area.

6.5.2 SYSTEMS ENGINEERING AGENT

MS2

Recommendation

Establish a policy that, for major new Weapon System programs, the Program Manager engage a Systems Engineering Agent to assist in problems arising in the translation of system requirements into detailed hardware and computer system design requirements. The agent, whether Government or contractor, should be highly experienced in system operational requirements, special purpose system hardware, and computer system software and hardware.

Primary Problem Areas Addressed

- Lack of Software Systems Engineering
- Inadequate Hardware/Software Tradeoffs
- Insufficient Software Definition
- Unrealistic Cost and Schedule Estimates

Discussion

The recommendations on Analysis and Validation of System Requirements (MP1), Systems Engineering of Computer Systems (SE1), and Computer Software (SE3) stressed the importance of a strong systems engineering approach to requirements, system architecture, and program configuration.

In view of the very complex nature of the operational environment of major Weapon Systems and the large scope of technical possibilities, these stages of system development require the highest level of technical expertise on the part of the customer as well as of the contractor. Even if experienced software specialists have been allocated to the Program Manager's staff, as suggested in Recommendation MS1, the amount of effort required in requirements analysis and review of hardware/software tradeoffs is such that one or two systems engineers cannot adequately represent the program manager in this area. These tasks require a strong group of engineers, with suitable analytical staff, computing and simulation facilities, and experience in systems engineering.

Whether or not a Systems Engineering Agent is required in a given program and, if so, whether such a group is an in-house organization, a Federal Contract Research Center (FCRC), or a contractor depends on the scope of the program, the capabilities of the Program Manager's staff, and the system expertise of the other organizations involved. Whatever the circumstances, however, it is most important that the systems engineering needs of the program be met. This consideration therefore should be an explicit point of management review.

A Systems Engineering Agent is needed in the conceptual and advanced development phases of a program, during which he may actually supervise the technical development. Such an agent is particularly valuable in

the preparation of the Request for Proposal (RFP), review of technical proposals for contractor selection, review of Computer Resources Development plans, and preparation for DSARC II reviews. He should actively participate in design reviews and in uncovering and resolving any critical design problems that arise during system development. He should assist in monitoring contractor progress and in reviewing test and integration plans. This activity should continue throughout system test and operational evaluation.

This need has been recognized in previous software management studies in various ways. For example, the Army Scientific Advisory Panel (Ref. 13) strongly recommended the use of an outside systems advisor to assist in program development.

Experience in Weapon Systems Programs. A number of large scale Weapon System Programs have designated an engineering or integration agent to oversee technical aspects of the development program. The Fleet Combat Direction Systems Support Activity (FCDSSA) (Dam Neck and San Diego) has acted in this role for many of the ship Combat Direction Systems, the CV development being an example. For older systems such as the DLG-28, the FCDSSA role has essentially been that of engineering agent for major system upgradings (such as the current development of the Naval Tactical Data Systems (NTDS) Model IV). FCDSSA has cited their knowledge of operational requirements as essential to program development. Their awareness of total requirements has enabled informal tradeoffs regarding performance and total life cycle costs. FCDSSA strongly recommended the designation of System Engineering Agents to support Program Managers throughout the acquisition of any Weapon System software.

The E-2C Program Manager used the NAVAIR Computer and Software System Group (Code 533) as technical support and review agent, as have a number of other NAVAIR Program Managers.

In the P-3C update program currently in progress, the Naval Air Development Center (NADC) is acting as design agent and is itself designing and developing the computer program.

In the AEGIS program, APL/JHU has functioned as a Systems Engineering Agent in its role of technical advisor to the Project Manager, and has carried out a substantial portion of the conceptual and program validation phases. RCA, the AEGIS system contractor, has provided system engineering direction for the software development.

#### Implementation

Include identification of a Systems Engineering Agent in the Program Management Plan, the Program Management Directive, and the Computer System Resource Development Plan. Provide for the agent's participation and, in appropriate cases, leadership in Concept Formulation and Program Validation, as well as in preparation for DSARC reviews, in the Request for Proposal process, and in Preliminary and Critical Design Reviews (PDR and CDR).

### 6.5.3 SOFTWARE OPERATIONAL SUPPORT AGENT

MS3

#### Recommendations

Require that a Software Operational Support (Maintenance) Agent be identified and consulted during the Program Validation Phase to support the Program Manager in providing for maintenance support requirements. Require that the agent be involved throughout Full Scale Development to plan for system integration, testing, and transfer from development to operational status.

#### Primary Problem Areas Addressed

- Unspecified Support Requirements
- Unspecified Maintenance Provisions
- Lack of Transferability

#### Discussion

The Navy has tasked several agencies with life-cycle maintenance responsibilities for various classes of systems. The Fleet Combat Direction Systems Support Activities (FCDSSA) at Dam Neck, Virginia, and San Diego, California, maintain shipboard Combat Direction Systems, and also maintain the avionics and weapon delivery systems for several classes of aircraft. The Naval Surface Weapons Center (NSWC) at Dahlgren, Virginia, maintains software for missile and gun fire control systems. The Naval Undersea Center (NUC) at San Diego, California, maintains underwater fire control systems for surface ships, and the Naval Underwater Systems Center (NUSC) at Newport, Rhode Island, maintains underwater combat systems for submarine platforms.

As designated support agents, these organizations have been involved, to varying degrees, in the conception and development of digital systems. The FCDSSA's have, in a number of cases, also acted as software system engineering agents with responsibility for development as well as for eventual support.

Software Configuration Control Boards. One important facet of FCDSSA and NSWC activity is their participation in Software Configuration Control Boards (SCCB). An SCCB for a given program is currently dissolved at the end of the Development Phase, that is, at termination of Shipbuilding and Conversion, Navy (SCN) funding for a ship construction program. In discussions with FCDSSA, Dam Neck, they recommended that the SCCB be strengthened during development and also continue, with less frequent meetings, throughout the life of each system. This would maintain positive and continuing control over system interfaces and would tend to prevent unnecessary program changes.

The SCCB for the DLGN-38 has centralized the configuration control of all five of the separately developed digital systems. It is chaired by FCDSSA, Dam Neck, and includes NSWC and other concerned agencies. It is responsible for implementation of design audits, Interface Design Specifications, and Software Engineering Change Proposals.

Experience in Weapon System Developments. FCDSSA, both Dam Neck and San Diego, emphasized the importance of a Maintenance Support Agent involvement during systems development. Examples were cited of systems that required extensive redesign or redocumentation because of inadequate preparation for compatibility with the maintenance activity. In other cases, such as in the E-2C Program, Navy involvement throughout the development phase resulted in smooth transition of software control from the contractor to the Navy support organizations.

Other approaches to system maintenance were noted. Trident operational support is in the planning stage, with the possibility of a new dedicated Trident maintenance facility being established. There are some programs such as Pershing for which operational support is provided by the original system contractor. The trend, however, appears to be to provide organizational operational support including maintenance capability in both the Army and Air Force, just as has been traditionally provided by the Navy. However, there is good reason to consider a degree of contractor participation in program maintenance on a long term basis, to make use of the contractor's detailed system knowledge as well as to facilitate system updating.

Requirements for Special Funding. While there is general agreement that a Software Operational Support Agent should be assigned early, there are practical obstacles to doing so. One obstacle that could be removed by management action is the lack of funds to support such activity at the proper time. RDT&E funds are always in short supply during Program Validation and at the outset of Full Scale Development, and there is strong pressure to meet development demonstration milestones. It is natural for Program Managers to delay funding activities that do not appear to be essential to initial goals. It is recommended that O&M funds be allocated for operation support involvement during Program Validation and Full Scale Development phases, not only to provide for the participation of the Support Agent, but also to fund contractor effort specifically devoted to the development of maintenance provisions.

#### Implementation

Amplify those parts of the Program Management Plan and the Program Management Directive dealing with the early participation of the Using and Supporting Commands to include the identification of an Operational Support Agent. Provide means for applying O&M funds to support contractor activity directed toward providing maintenance capabilities and documentation.

6.6 ACQUISITION MANAGEMENT STANDARDS

6.6.1 STANDARD CRITERIA FOR WEAPON SYSTEMS COMPUTER RESOURCES  
ACQUISITION MANAGEMENT

AM1

Recommendation

Establish a common set of requirements and criteria to be applied in the acquisition and support of Weapon Systems computer resources by all services.

Primary Problem Areas Addressed

Undeveloped Design and Control Technology  
Lack of Policy Guidance and Planning  
Inadequate Management Procedures  
Lack of Established Standards

Discussion

The subject of software standards is an extremely complex one, especially in view of the recent efforts on the part of all services to provide a higher degree of uniformity and control to software acquisition. A good summary of the status of software standards as of 1974 is contained in a paper by Wolverton in the Aeronautical Systems Software Workshop (Ref. 25).

The standards and procedures reviewed in the DoD Software Study by APL included the following:

DoD 5000.1	Acquisition of Major Defense Systems, July 1971 (Ref. 43)
Dod 5010.19	Configuration Management, July 1968 (Ref. 35)
DoD 4120.17M	Automated Data System Documentation Standards Manual, December 1972 (Ref. 44)
MIL-STD-881	Work Breakdown Structures for Defense Material Items, November 1968 (Ref. 5)
MIL-STD-483	Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs, June 1971 (Ref. 9)
MIL-STD-490	Specification Practices, May 1972 (Ref. 8)
MIL-STD-499A	Engineering Management, May 1974 (Ref. 21)
MIL-S-52779 (AD)	Software Quality Assurance Program Requirements, April 1974 (Ref. 17)
SSD 61-47B	Computer Program Subsystem Development Milestones, April 1966 (Ref. 3)
AFR-800-14, Vol. I	Management of Computer Resources in Systems, May 1974 (Ref. 2)
AFR-800-14, Vol. II	Acquisition and Support of Computer Resources in Systems, 1975 (Ref. 10)
WS-8506	Requirements for Digital Computer Program Documentation, December 1966 (Ref. 28)
SECNAVINST 3560.1	Tactical Digital Systems Documentation Standards, August 1974 (Ref. 33)

SECNAVINST 5000.1      System Acquisition in the Department of the Navy,  
March 1972 (Ref. 45)  
NAVMATINST 4130.1A      Configuration Management, July 1974 (Ref. 18)

In addition the recently prepared Army AMC Pamphlet AMCP-70-4 (Research and Development Software Acquisition - A Guide for the Materiel Developer, Ref. 19), and the Navy Computer Software Management Task Force Document Outlines (Ref. 20) were examined.

While there is much discussion about "proliferation" of software standards, much of this is directed to documentation or specification practices and formats, as opposed to acquisition management procedures. In the latter category, the only comprehensive official document dealing specifically with software acquisition management is the very recently published AFR-800-14 (Refs. 2 and 10). The Army-developed MIL-S-52779 (AD) (Ref. 17) provides good supplementary material on Quality Assurance. MIL-STD-499A (Ref. 21) is strictly hardware oriented, and MIL-STD-483 (Ref. 9) is largely so.

#### Implementation

Derive a tri-service document covering the procedures to be used in the acquisition and support of Weapon Systems computer resources, using current service regulations and manuals as a basis. Such a document would most appropriately become a Military Standard. It is suggested that Air Force Regulation (AFR) 800-14, Vol. II (Ref. 10), be used as a point of departure, amended as recommended herein and to be consistent with MIL-S-52779 (AD) (Ref. 17) and NAVMATINST 4130.1A (Ref. 18). Additional material may be drawn from the Army's AMC Pamphlet 70-4 (Ref. 19) and the Navy's Computer Software Management Task Force Document Outlines (Ref. 20). Use a common terminology along the lines recommended by the Joint Logistics Commanders' Software Reliability Work Group (see Section 5.1.4).

An interim approach would be to make modifications to AFR-800-14 (Refs. 2 and 10) to make it acceptable to all services and designate it a Military Standard. Then supplementary documents, tailored to each service's particular needs, might be developed and issued to embody the more far-reaching modifications and additions required by each service.

## 6.6.2 SOFTWARE ACQUISITION GUIDES

AM2

### Recommendation

Prepare a series of handbooks or guides covering important aspects of software acquisition, to help Program Managers and their staffs to define, review, and evaluate requirements, procedures, proposals, and designs during precontract and contract management. These would include such items as:

1. Life Cycle Plan
2. System Requirements Review
3. Request for Proposal (RFP) Preparation and Review
4. Computer Resource Development Plan Review
5. Preliminary and Critical Design Reviews
6. Documentation Standard Selection
7. Support Facility Plan Evaluation
8. Quality Assurance (QA) Plan Evaluation

### Primary Problem Areas Addressed

Insufficient Understanding by Managers  
Undeveloped Design and Control Technology  
Lack of Policy Guidance and Planning

### Discussion

It is neither practical nor desirable to develop official standards to govern all aspects of software acquisition management. The variation in nature, scope, and organization of different Weapon Systems and the particular needs of the individual services necessitate substantial flexibility to enable Program Managers to direct their programs in a cost-effective manner in accordance with the policy of DoD 5000.1 (Ref. 43). At the same time, the Program Manager's capabilities are peculiarly limited in the area of software acquisition owing to the relatively undeveloped state of management methodology and the limited number of qualified staff personnel experienced in this area. This makes it important to provide Program Managers with a series of clear and concise handbooks or guides. In doing so, care must be taken to avoid unnecessary proliferation of documents that are mutually inconsistent and nonuniform in terminology.

Industry and Weapon System Software Manuals. In visiting prominent software contractors, it was found that each had a well-developed set of manuals for internal use covering planning, design, implementation, and testing. While some treated the manuals as proprietary data, others freely distributed their literature. Of the latter, the TRW Software Development and Configuration Management Manual (Ref. 41) was found to be particularly complete and clearly written. Figure 6-7, taken from the TRW Manual, is a highly illuminating diagram showing the interrelations between the phases of the system life cycle and the various reviews, products, and documents.

The limited guidance provided by DoD and service standards and procedures has led to the generation of individual manuals for some Weapon Systems.



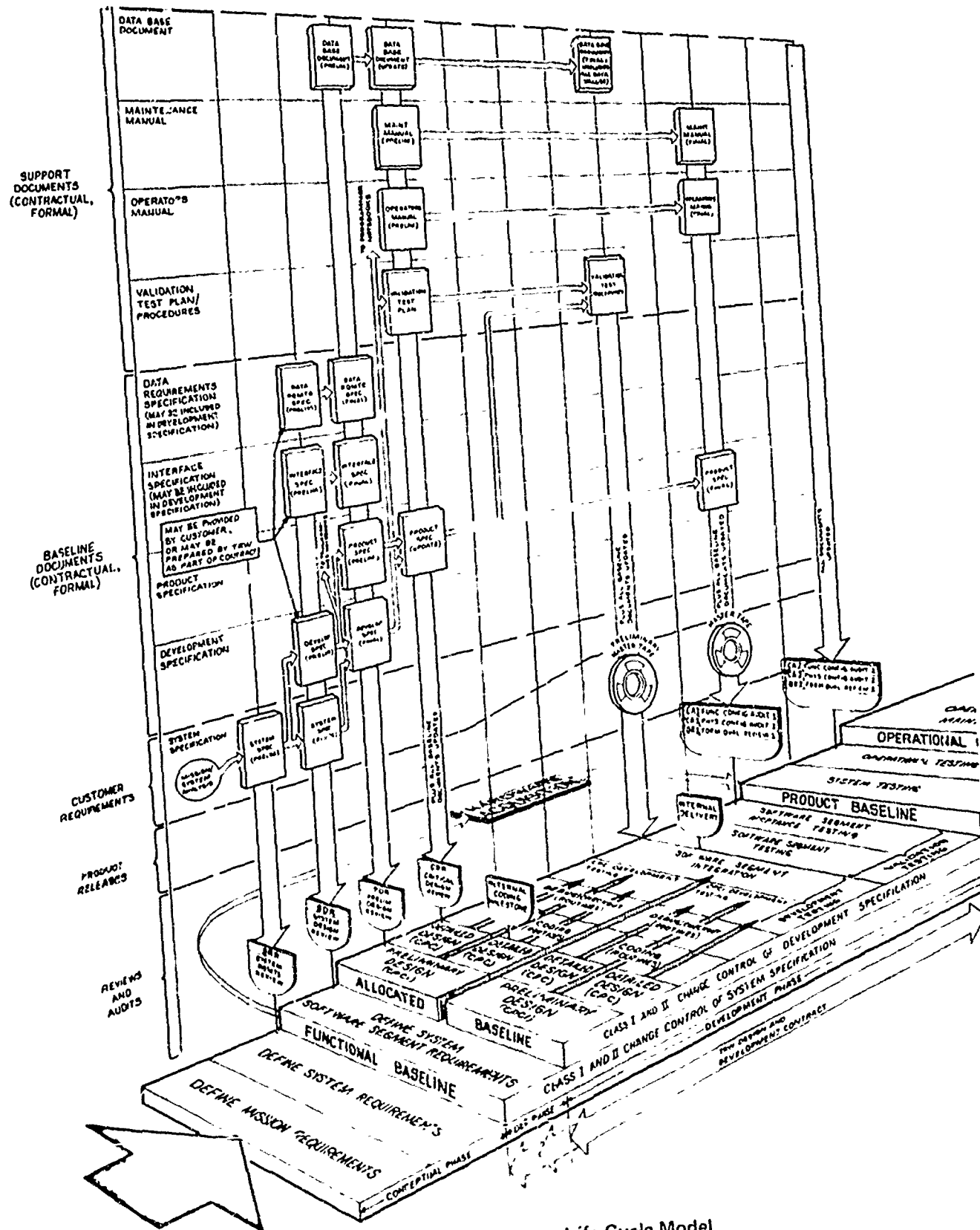


Fig. 6-7 Partial System Life Cycle Model

For example, the development effort for the DLGN-38 generated its own Software Configuration Control Manual, which expands on MIL-STD-480 (Ref. 30) and NAVMATINST 4130.1A (Ref. 18). In addition, the SAM-D Program Manager's Office indicated that manuals covering all phases of the software acquisition process would be very useful.

Guides to Applicable Documentation. One of the most troublesome problems confronting Program Managers is the type and extent of documentation appropriate to a given software program. The initial cost of documentation is high, but life cycle cost is very sensitive to lack of proper documentation. WS-8506 (Ref. 28) and its successor SECNAVINST 3560.1 (Ref. 33) give good descriptions of the various types of documentation that may be required. However, the choice of which documents should be required for each type of software program is left open.

DoD Automated Data System Documentation Standards Manual 4120.17M (Ref. 44) describes a useful method for determining the appropriate level of documentation as a function of five levels of complexity. The level for a particular program is derived by rating 12 complexity factors on a scale of one to five. The 12 factors, which include such items as "change in scope and objective," "personnel assigned," "criticality," and "concurrent software development," give a comprehensive basis for the overall rating.

An in-depth study on documentation standards was carried out by the Systems Development Corporation for the Naval Electronics Laboratory Center and was published in 1973 under the title "Software Milestone Measurement Study" (Ref. 46). This study compared a number of documentation standards as a function of milestone documents. It also included a very complete discussion of the criticality of each type of document as a function of project size and risk. The discussion and associated tabulation represent a worthwhile guide for selecting program documentation.

A troublesome factor in examining documentation standards is the difference in names for various types of documents, and the differences in requirements for those seemingly intended for the same purpose. These differences were stated by industrial contractors to result in substantial added costs, and their reduction represents a high payoff area.

JLC SRWG Handbooks. The Joint Logistics Commanders' Software Reliability Work Group has recommended the preparation of tri-service handbooks. Their preliminary list includes items under the following headings:

- Formulating a Life Cycle Plan
- Specification and Contracting
- Development Visibility and Control
- Product Control
- Quality Assurance
- Maintenance
- Regulations, Specifications, and Standards

A coordinated effort would have many advantages in achieving uniform terminology and maximum yield.

THE JOHNS HOPKINS UNIVERSITY  
APPLIED PHYSICS LABORATORY  
LAUREL, MARYLAND

Implementation

Coordinate current service efforts or assemble a tri-service committee with Government and industry representation, under the sponsorship of the Office of the Secretary of Defense, to prepare suitable handbooks. Issue drafts for interim guidance and to obtain feedback from experience. Allocate special funds to participating service agencies.

## 6.7 DEVELOPMENT OF TOOLS AND TECHNIQUES

### 6.7.1 SOFTWARE TEST TOOLS

TT1

Support development of improved software test and validation tools to reduce the cost and time involved in software verification. These should include automated tools to identify and exercise all branches, to detect and isolate design faults, and to categorize error sources.

#### Primary Problem Areas Addressed

Cost of Validation and Modification

#### Discussion

Earlier recommendations highlighted the importance of software tools and the need to specify the utilization and deliverability of existing tools. This recommendation deals with the need for DoD support of R&D in this important area.

Need for Test Tools. Weapon system software programs with hundreds of subroutines and hundreds of thousands of instructions are becoming common. Verification (agreement with specification and design) and validation (performance in the operational environment) of this complex software are central to the effectiveness of the total system. When the magnitude and criticality of verification and validation are considered, it becomes clear that the only viable solution is one that uses the power of the computer to solve some of the problems it has created.

The trend is toward integrated tools rather than the earlier ad hoc collections of unrelated and one-shot techniques. B. C. DeRoze in his paper to the Aeronautical Systems Software Workshop (Ref. 25) discusses a "global" approach to software methodology which involves an "anticipatory rather than a reactionary process." He advocates the preventive rather than the curative concept of testing. Stucki (in another paper from Ref. 25) concludes that "we must design with verification and validation constantly in mind... and continue to develop an integrated set of automated support tools and a management discipline requiring their use and refinement." The Monterey Symposium on the high cost of software (Ref. 26) calls for R&D to develop the theory and effective methods for formal verification and proof of program properties. It suggests that research in computer systems be strengthened and coupled closely to software research. The Army Scientific Advisory Panel study (Ref. 13) emphasizes the need for "Improved methods for specifying, selecting, developing, testing and evaluating tactical hardware and software."

Adoption to Weapon Systems. Many test tools have been developed by contractors for nonmilitary usage. A recent survey done for the Air Force identified twelve different tools specifically for testing computer programs. Nine of these, which are available commercially, are tabulated on the following page.

THE JOHNS HOPKINS UNIVERSITY  
APPLIED PHYSICS LABORATORY  
LAUREL, MARYLAND

<u>Contractor</u>	<u>Test Tool</u>
Computer Software Analysts, Inc.	QUALIFIER
TRW Systems Group, Inc.	PAGE NODAL
General Research Corp.	RSVF SOFTOOL
McDonnell Douglas	PET
NASA	TDEM
General Electric Co.	ZYGO
CAPEX	FORTUNE

At present most of these tools can test Fortran programs but none as yet has been extended to test Weapon System programs, which are usually written in either JOVIAL, CMS-2, TACPOL or assembly language. Many of these tools can be made available for Weapon Systems by rewriting them to operate on programs written in the above high order languages. In fact, recently some of these contractors have been funded by the Air Force to extend their tools to test JOVIAL programs.

Support of Advanced Tools. Every software contractor interviewed highlighted the need for R&D support in the area of tools. The areas of proof-of-correctness, automated selection of test cases, path segment analysis, exercising of software capabilities, and test drivers which model the external environment are a small sample of fields related to software testing that need further development. Many of the existing tools mentioned earlier need to be expanded to encompass these advanced techniques.

In addition, GRC recommends DoD support of R&D for the development of comprehensive test cases. The Information Sciences Institute of the University of Southern California calls for support of the National Software Works (NSW). This R&E project, which is discussed in more detail in Section 5.2.4, plans to set up a software tools clearinghouse and use the ARPANET to make these aids available to users throughout the country. The advanced verification and validation work being conducted and sponsored by RADC also deserves support.

TRW estimates that 40% of software costs are in the test and integration phase, and Boeing feels the figure may be closer to 50%. Thus, software test tool development is considered one of the highest potential technical pay-off areas in software acquisition and deserves the encouragement and support of the Department of Defense.

Implementation

Support ongoing service programs in development of automated test and validation tools. Fund the conversion of selected tools to the high level languages used in Weapon Systems (e.g., CMS-2, JOVIAL) and provide them to system contractors and Operational Support Activities as soon as economically practicable. Invite innovative proposals for new work. Support R&D efforts in software portability to aid in the application of tools to different systems.

## 7. SUBJECTS FOR FURTHER INVESTIGATION

During the course of this software study, a number of areas and techniques with potential future payoffs were identified. Most of these areas have not been examined in sufficient depth for APL to make specific recommendations. They have been studied sufficiently, however, to establish that they have a definite bearing on the cost and performance of Weapon Systems software and should be considered as candidates for possible further investigation. The subjects are listed below.

### Information Exchange

The obvious need for extensive exchange of information on computer hardware and software RDT&E technology has inspired many symposia, societies, associations, and other activities. There appears to be merit in DoD sponsorship of formally constituted and continuing information exchange, both among the Services and between DoD and industry.

### Coordination of Standards and Terminology

The inconsistency of software standards and terminology among DoD and Service publications is a widely recognized problem in software acquisition. Recommendations AM1 and AM2, Section 2, addressed those standards that are concerned with acquisition management and planning, but did not treat lack of consistency in other types of standards and in terminology.

### Software Costs

The main impetus of this study has been on attacking the root causes of undue software costs rather than on the development of cost investigative procedures. In addition, Recommendations MP2 and MP3 are directed to increasing the visibility of software costs, and thus bringing them under greater management control, as well as providing cost experience applicable to future programs. Beyond these measures, there appears to be a payoff in developing better means for estimating software costs.

### Education in Computer Systems Engineering

There is a widespread shortage of well-trained computer system engineers, which has particular impact on Weapon System software development. At present, educational institutions do not offer programs of instruction that produce such individuals, and hence their supply is likely to remain critically short. An innovative approach to a combination of education and training appears to be required.

### Application of Commercial Developments to Weapon Systems

Equipment and software now commercially available or being developed may be advantageous for military application. This objective runs counter to those of standardization for logistic and training purposes, as well as certain legal and proprietary restrictions.

### Integration of Software Developed by Multiple Contractors

Both the expanding use of software and the increased use of smaller processors lead to systems in which the development may be allocated among multiple contractors. Current specification practices do not always ensure compatible development in such instances.

### Techniques for Sizing Software Tasks

Uncertainties in initial estimates of software implementation requirements have far-reaching impacts on system development planning. Improved sizing methods are needed to support requirement analysis and resource allocation.

### Software Portability

The ability to apply developed computer software, especially support software, to new and different systems may provide substantial cost savings. Requirements and techniques for achieving such portability are now generally lacking and little new development work applicable to Weapon Systems was found in this area.

### Software Requirements Analysis Methods

Better methods are needed to analyze requirements for consistency and to assess their impact on computer software size and complexity. The importance of this analysis during the Program Validation Phase was addressed in Recommendation MP1. Promising research has recently been initiated in this area.

### Compilers for Higher Order Languages (HOL's)

An HOL usually has several compilers, each for different processors or dialects of the language. These compilers have in the past taken considerable time to implement and have in several cases delayed the availability of new Weapon Systems. Work is underway at several organizations to develop new methods for validating compilers and for implementing portions of them in a manner independent of the host and target processors.

### Exploitation of Microprocessor Technology

Microprocessor technology is experiencing rapid growth and offers many benefits if applied to Weapon System designs. Exploitation of this technology may require new approaches to developing software.

### Graphical Design Representations

Several different representations and formats currently are used to depict the different stages of system development. As a result, it is generally impossible to determine whether or not each stage of definition, design, and implementation truly represents a translation from the next higher level specification. Several promising graphical approaches to this important problem are being developed.



#### Automated Programming

The design and implementation phases of software development are currently manpower intensive and prone to error. The potential exists for automating some of the steps of implementing computer programs from engineering-level design statements.

#### Casualty Modes

Modes that provide continued operations in case of computer equipment failure are missing in some systems; other systems require a break in operation to load predefined program subsets. Current trends in computer system design enable the development of systems that "degrade gracefully" with equipment failure. Such systems should provide for human control and dynamic reallocation of remaining resources to match immediate operational needs.

### 9. GUIDE TO APPENDICES

#### 8.1 FINDINGS AND RECOMMENDATIONS OF PREVIOUS STUDIES

Appendix A, Findings and Recommendations of Previous Studies, contains material extracted and/or summarized from the ten designated Baseline Documents. More detail is included than in the discussion given in Section 3 of this volume.

The ten designated Baseline Documents are:

Electronics-X: A Study of Military Electronics with Particular Reference to Cost and Reliability

Tactical Computer Software Acquisition and Maintenance Staff Study

Report of the Army Scientific Advisory Panel Ad Hoc Committee for Army Tactical Data System Software Development

Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980s (CCIP-85)

Project Pacer Flash

Automatic Data Processing Costs in the Defense Department

A Report on Air Force Logistics Command Operation Flight Program Support

Proceedings of the Aeronautical Systems Software Workshop

Proceedings of a Symposium on the High Cost of Software Held at the Naval Postgraduate School, Monterey, California, on September 17-19, 1973

Government/Industry Software Sizing and Costing Workshop

A brief introduction specifying the purpose of each study and a summary of its findings and/or conclusions are included. Recommendations are summarized for each study that provided them. Whenever such study recommendations are available, abbreviated versions of the APL recommendations (from this report) that correlate most closely are included for reference.

## 8.2 WEAPON SYSTEM STUDIES

These appendices present supplemental information on each of the Weapon Systems that were reviewed in this study (Section 4 of this volume). The systems are:

Appendix B, Shipborne Systems

- DLG-28 Combat System
- DDG-9 Combat System
- DLGN-38 Combat System
- AEGLS Weapon System
- CV Tactical Data System

Appendix C, Airborne Systems

- E-2C Tactical Data System
- P-3C Airborne Patrol System
- S-3A Airborne Weapon System
- F-14 Avionics and Weapon Delivery System

Appendix D, Undersea and Landbased Systems

- Trident Command and Control System
- Pershing Weapon System
- SAM-D Weapon System

An overview is given for each Weapon System including a general system description, the architecture of the embedded computer system, and the architecture of the computer software, to the extent that this information was available. This is followed by sections on software definition, design, and implementation; software validation and integration; software acquisition management organization and methods; and operational software maintenance. Pertinent points from these discussions are highlighted in Section 4 of this report.

The intent of this overview is to provide an understanding of the scope, basic requirements, and types of Weapon Systems computer software that are now being produced and used operationally.

A discussion of the approach to life-cycle maintenance (when available) is presented for Weapon Systems that are deployed, or for which maintenance plans have been formulated.

### 8.3 BIBLIOGRAPHY

Appendix E, Bibliography, contains a bibliography of the relevant documents that were acquired and used throughout the course of this DoD Software Management Study. The majority of these documents are generally available.

A Software Library was established to support the study effort by providing easy access to necessary background material. A short discussion of the methods used to acquire, process, and circulate the library documents is included. Various tools such as computer-generated indexes and batch and on-line searching provided rapid retrieval of desired information. This Software Library system will be maintained to support on-going and future software efforts at the Applied Physics Laboratory.

#### REFERENCES

1. M. R. Currie, A. I. Mendolia, and T. E. McClary, Management of Weapon System Software, Office of the Secretary of Defense, Washington, D.C., 3 December 1974.
2. Acquisition Management, Management of Computer Resources in Systems, AFR-800-14, Department of the Air Force, Washington, D.C., 10 May 1974.
3. Computer Program Subsystem Development Milestones SSD Exhibit 61-47B, Department of the Air Force, Washington, D.C., 1 April 1966.
4. Defense Systems Acquisition Review Council (DSARC), DoD-5000, Department of Defense, Washington, D.C.
5. Military Standard - Work Breakdown Structures for Defense Material Items, MIL-STD-881, Air Force Systems Command, Washington, D.C., 1 November 1968.
6. E. T. Reich, Tactical Computer Software Acquisition and Maintenance Staff Study, Office of the Assistant Secretary of Defense, Washington, D.C., 31 October 1973.
7. Project Pacer Flash - Volume I. Executive Study and Final Report, Air Force Logistics Command, Wright Patterson AFB, Ohio, 28 September 1973.
8. Military Standard - Specification Practices, MIL-STD-490, Department of Defense, Washington, D.C., 18 May 1972.
9. Military Standard - Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs, MIL-STD-483, Air Force Systems Command, Washington, D.C., 1 June 1971.
10. Acquisition Management. Acquisition and Support of Computer Resources in Systems, Vol. II, AFR-800-14, Department of the Air Force, Washington, D.C., 1975.
11. Handbook for Program Development and Production Procedures of Digital Processor Program, H(A)-4010, Fleet Combat Direction Systems Support Activity, Dam Neck, Va., 12 July 1974.
12. H. P. Gates, Jr., B. S. Gougary, and S. J. Deitchman, Electronics-X: A Study of Military Electronics with Particular Reference to Cost and Reliability. Volume 2: Complete Report, R-195, Institute for Defense Analyses, Arlington, Va. January 1974.

13. Report of the Army Scientific Advisory Panel Ad Hoc Committee for Army Tactical Data System Software Development, Army Scientific Advisory Panel, Washington, D.C., October 1974.
14. Standard Reserve Capacity Requirements for Digital Combat System Processors, TADSTAND-5, Naval Material Command, Washington, D.C., 9 May 1972.
15. A Report on Air Force Logistics Command Operation Flight Program Support, Vol. I, Introduction and Summary, TM-5439/000/00, System Development Corporation, Santa Monica, Cal., 9 December 1974.
16. A Report on Air Force Logistics Command Operation Flight Program Support, Vol. II. Data, Analysis and Conclusions, TM-5439/001/00, System Development Corporation, Santa Monica, Cal., 9 December 1974.
17. Military Specification - Software Quality Assurance Program Requirements, MIL-S-52779 (AD), Department of the Army, Washington, D.C., 5 April 1974.
18. Configuration Management, NAVMATINST 4130.1A; AR-70-37; Department of the Navy, Washington, D.C., 1 July 1974.
19. Research and Development Software Acquisition - A Guide for the Materiel Developer, AMCP-70-4, Army Materiel Command, Alexandria, Va., 2 September 1972.
20. Computer Software Management Task Force Document Outlines, Department of the Navy, Washington, D.C., 15 January 1975.
21. Engineering Management, MIL-STD-499A, Air Force Systems Command, Washington, D.C., 1 May 1974.
22. Information Processing/Data Automation Implications of Air Force Command and Control Requirements in the 1980s (CCIP-85), SAMSC-XRS-71-1, Executive Summary (rev. ed.), SAMSO TR 72-122, and Vol. I, Highlights, SAMSO TR 72-141, USAF Space and Missile Systems Organization, Los Angeles, Cal., 1972.
23. Computer Software Support for Weapon Systems, AFR-20-1, Department of the Air Force, Washington, D.C., 4 December 1972.
24. D. A. Fisher, Jr., Automatic Data Processing Costs in the Defense Department, IDA-P-1046, Institute for Defense Analyses, Arlington, Va., October 1974.
25. Proceedings of the Aeronautical Systems Software Workshop, AD-A004547, Air Force Systems Command, Washington, D.C., April 1974.

26. Proceedings of a Symposium on the High Cost of Software Held at the Naval Postgraduate School, Monterey California, on September 17-19, 1973 (The High Cost of Software), AD-777-121, SRI Project 3272, Stanford Research Institute, Menlo Park, Cal., September 1973.
27. Summary Notes of a Government/Industry Sizing and Costing Workshop, Electronic Systems Division, Hanscom AFB, Mass., 11 February 1975.
28. Requirements for Digital Computer Program Documentation, WS-8506, Naval Ordnance Systems Command, Washington, D.C., 15 December 1966.
29. Integrated Combat System Management Plan, Revision 5, Naval Sea Systems Command, Washington, D.C., January 1975.
30. Military Standard-Configuration Control Engineering Changes, Deviations and Waivers, MIL-STD-480, Naval Air Systems Command, Washington, D.C., October 1968.
31. Software Configuration Control Procedures Manual, NAVSEA 0900-LP-080-2010, Naval Sea Systems Command, Washington, D.C., 2 March 1975.
32. Computer Program Development Plan, AEGIS Weapon System CDRL Sequence No. 178, RCA Corporation, 1 April 1972.
33. Tactical Digital Systems Documentation Standards, SECNAVINST 3560.1, Department of the Navy, Washington, D.C., 8 August 1974.
34. Trident CCS Software Management Plan, NAVSHIPS 0900-075-4010, Department of the Navy, Washington, D.C., 1972.
35. Configuration Management, DoD 5010.19, DDR&E, Washington, D.C., July 1968.
36. Configuration Management Implementation Guide, DoD 5010.21, DDR&E, Washington, D.C., August 1968.
37. Armed Services Procurement Regulation, ASPR1-202(a), Department of Defense, 1975.
38. Construction and Control of Digital Processor Programs for the Navy Combat Direction Systems, OPNAVINST 3500.27B, Office of the Chief of Naval Operations, Washington, D.C., 28 June 1974.
39. [FCDSSA,] San Diego and [FCPSSA,] Dam Neck... modification of mission of, OPNAVNOTE 5450, Chief of Naval Operations, Washington, D.C., 22 December 1972.

40. Technical Management and Life Cycle Budgeting of Digital Computer Data System Software, Draft, H-4072, Fleet Combat Direction Systems Support Activity, San Diego, Cal., June 1975.
41. B. W. Boehm, Software Development and Configuration Management Manual, TRW-SS-73-07, TRW Systems Group, Redondo Beach, Cal., 17 December 1973.
42. Proceedings of the TRW Symposium on Reliable, Cost-Effective, Secure Software, TRW SS-74-14, TRW Systems Group, Redondo Beach, Cal., March 1974.
43. Acquisition of Major Defense Systems, DoD 5000.1, DDR&E, Washington, D.C., 13 July 1971.
44. Automated Data System Documentation Standards Manual, DoD Manual 4120.17M, Department of Defense, OASD (Comptroller), Washington, D.C., 29 December 1972.
45. System Acquisition in the Department of the Navy, SEGNAVINST 5000.1, Department of the Navy, Washington, D.C., 13 March 1972.
46. N. S. Mathis and N. E. Willmorth, Software Milestone Measurement Study, NELC TD 285, AD-775305, Naval Electronics Laboratory Center, San Diego, Cal., 7 November 1973.





OFFICE OF THE SECRETARY OF DEFENSE  
WASHINGTON, D.C. 20301

3 DEC 1974

MEMORANDUM FOR The Assistant Secretary of the Army  
(Installations and Logistics)  
The Assistant Secretary of the Navy  
(Installations and Logistics)  
The Assistant Secretary of the Air Force  
(Installations and Logistics)  
The Assistant Secretary of the Army  
(Research and Development)  
The Assistant Secretary of the Navy  
(Research and Development)  
The Assistant Secretary of the Air Force  
(Research and Development)

SUBJECT: Management of Weapon System Software

The sharply rising costs of software programs in the weapon system acquisition process, with respect to acquisition procedures, development and maintenance of such software, and the increasing importance of the software role in the overall mission effectiveness of major DoD weapon systems constitute serious technical and management problems that must be solved if we are to have the weapon systems that are needed for our national security. To find solutions to these problems, we are initiating a two phase study program which will require the joint involvement of the OSD staff and the Services.

The first phase of the study program is only now starting. Its major effort centers on two four month studies by the Mitre Corporation and the Applied Physics Laboratory at Johns Hopkins University to identify and define (1) the nature of the critical software problems facing the DoD, (2) the principal factors contributing to the problems, (3) the high pay-off areas and alternatives available, and (4) the management instruments and policies that are needed to define and bound the functions, responsibilities and mission areas of weapon systems software management. The second phase of the study program will be to examine in depth those areas which have been surfaced in the first phase as having first-order importance to the DoD. It is not

unlikely that a study group will be organized at this time having the following objectives: Identify and evaluate current and alternative Defense and commercial software policies and practices in development, procurement and operational support which most significantly influence acquisition and life cycle costs, field reliability, maintenance, standardization and to identify possible improvements to reduce and control costs and improve software reliability, standardization, maintainability and software research and development production capabilities.

The software study program needs direct service participation by military officers or civilian experts experienced in requirements generation, weapon systems acquisition, support and management techniques as they apply to software. In the first phase of the effort these needs can best be met by having two individuals from each Service identified to serve on the Software Steering Committee. It is recommended that one individual have an R&D background and the other have logistics experience. It is not anticipated that the services of the individuals identified will be required on a full time basis.

The Committee will assist in developing the study goals for each phase of the total effort, provide focal points within the DoD to coordinate and support the study objectives, assist in obtaining the data needed in accomplishing the studies and to make recommendations on how to implement study findings and to determine the nature and extent of the follow-on activities. It is suggested that personnel at the O-6 level be considered for assignment to the Committee and that they be selected on the basis of the Committee's needs, responsibilities and objectives as outlined above.

The first meeting of the Software Steering Committee, with the contractors, is planned for Friday, 13 December at 1330 hours in Conference Room 1E 801 #4. You are requested to have the names of your committee representatives to Col. R. D. Hensley, OASD(I&L)WA, Room 2A 318 prior to this date.


  
MALCOLM R. CURRIE

Director of Defense  
Research and Engineering

  
ARTHUR I. MENDOLIA

Assistant Secretary of Defense  
(Installations and Logistics)

*Concur  
@ Luck*

  
TERENCE E. McCLARY  
Assistant Secretary of Defense (Comptroller)